**TECHNICAL DESIGN DOCUMENT**

Contents

[Random notes I took from library books 9](#_Toc173855273)

[Player 11](#_Toc173855274)

[Ally 12](#_Toc173855275)

[Enemies 12](#_Toc173855276)

[Advanced enemy 15](#_Toc173855277)

[Enemy categories 15](#_Toc173855278)

[Entity 15](#_Toc173855279)

[Entity manager 15](#_Toc173855280)

[Class changer 18](#_Toc173855281)

[Inventory 22](#_Toc173855282)

[weapons 24](#_Toc173855283)

[UI 25](#_Toc173855284)

## Random notes I took from library books

A sense of presence is created through a combination of operator’s actions and the subsequent video, audio, and haptic feedback.

Most interactions with an environment are possible because we have an internalized knowledge of how various aspects of that environment work. When we are faced with experiences we cannot readily interpret, our mode of being becomes more critically removed, and we must actively think about what we are doing.

Prior experience, expectations, and knowledge form a crucial part of this interpretative relation.

Six characterizations of presence:

* presence as social richness
* presence as realism
* presence as transportation
* presence as immersion
  + perceptual immersion
  + psychological immersion
* presence as social actor within a medium
* presence medium as social actor

common amongst all of them: define presence as “the perceptual illusion of non-mediation”

#### immersion

by engaging with the game world as a map rather than a spatial environment, the player remains in conceptual rather than inhabited space.

* Game idea: since is first person, player cant see own face. “how does own model look like”, player might think that way. Make game around the term “curiosity kills the cat”. It was protagonist’s curiosity that caused his situation to spiral to game setting (abandoned at post-apocalyptic ruins). What if player has been trying his best to find a cure out of his own kindness only to find out he is abandoned because he was infected, suddenly finding a cure becomes a necessity.

#### Player involvement model

A distinction between aspects of a game which engages player in the moment of playing from aspects that attracted players to the game initially and kept them returning to the game over time. The dimensions are experienced not in isolation, but always in relation to each other. They are experienced unconsciously during the interpretative and communicative process, and therefore, play an important role in noticing and directing attention towards aspects of a given reality. This reality is made up primarily of stimuli originating from the game environment, but at times the stimuli originate from the surrounding physical environment.

#### Macro and micro involvement

Macro – factors that shape player’s opinion and disposition towards the game prior and following the game experience.

* E.g. feelings, expectations

It concerns issues of motivation and sustained engagement with games through the long-term aspects of the six dimensions of involvement.

Micro – moment by moment engagement of gameplay.

Make a distinction between the general direction of attention towards a medium and the form of active involvement prevalent during gameplay.

Most of the time, we are not aware of the way which attention affects our performance or behaviour. It becomes more apparent when we are trying to comprehend complex information, learn new tasks, or engage in activities unfamiliar to us. In such situations, the information required to solve the task or manage the situation can be greater than what our attention capacity system can handle.

Six dimensions of player involvement model

* Shared involvement
  + Engagement derived from players’ awareness and interaction with other agents in a game environment.
  + Encompasses all aspects relating to being with other entities in a common environment.
* Narrative involvement
  + Engagement with story elements that have been written into a game, as well as those that emerge from players’ interaction with the game.
* Affective involvement
  + Encompass various forms of emotional engagement.
  + Accounts for the rhetorical strategies of effect
* Spatial involvement
* Kinaesthetic involvement
* Ludic involvement

#### Core gameplay loop

Repetition is one of the fundamentals of play. When people enjoy something, they want to repeat it.

* Video games have a few selected mechanics repeat over and over throughout the experience.
* Consists of specific verbs – steering, sprinting or sneaking etc.

Loop of core gameplay loop comprises an action on player’s part. The result of that action, the player’s reaction to the result, and the game requiring the player to repeat the original action to progress.

Although many people may consider “repetitive” to be a negative word, the art of creating video games must involve embracing repetition and providing circumstances that keep it interesting, compelling and rewarding.

Tips for designing actions:

* Easily understood
* Easily performed
* Enjoyable
* Able to provide direct feedback
* Flexible enough for multiple scenarios
* Extendable with additional actions
* Combinable with other actions
* Evolvable to provide support of other player loops

#### Feedback loop

Types of feedback loop:

* Positive feedback loop – achieving a goal is rewarded, which makes it easier to continue achieving goals.
* Negative feedback loop – achieving a goal makes it harder to continue achieving goals.

Players are generally only interested in rewards that help them win. Although players want to be more powerful, what they really want is an interesting and challenging game. Sometimes, giving players’ more power goes counter to that goal.

Negative feedback loops are sometimes seen as unfair. One must be careful in implementing negative feedback loops so that the performance in the game isn’t seen as irrelevant.

* E.g. rubber banding mechanic in racing games, it acts a reward for bad driving and appears unfair for proficient driver.

How to fix a negative feedback loop:

Reward the player for achieving the game’s goals.

How to fix positive feedback loop:

Designers may want to scrap positive feedback entirely, but they must be careful to make sure the players still feel an intrinsic reward for completing the game’s goals. Alternatively, consider finding a reward that doesn’t contribute to the player’s efficacy, like cosmetic rewards.

## Player

Health: 100hp

Default weapon: knife

Movement speed: 50 (placeholder value lol)

**Some notes for self:**

Player need to be mindful of bullet resource (maybe only civilian enemies will drop bullets but not a lot oso)

Player can probably only control 4 enemies at first

After mechanic upgrade, player can control 6 regular enemies, or 1 advanced enemy and 1 regular enemy

Player need to protect themselves. When players are free from enemies, they can revive an enemy to be own ally

Current tasks:

Knife – weapon

Enemy – attacks player when see

Ally – enemy but friendly

Game system – survive as long possible bs

Loot chests – drop from sky

menu/ pause screen

## Ally

Enemy after player use resurrection mechanic. Will follow player around and shoot nearby enemies. Will still have same max health and attack damage as when it is enemy.

Since there is more than 1 enemy type, ally will need to be able to copy all related details from enemy without losing out anything. Some parts not compatible to gameplay might be removed and replaced with something else.

## Enemies

Weak Point:

player can shoot at enemies weak point to deal 1.5x dmg

Universal enemy detection:

* if player within enemy sight, target player
* If player move around enemy and is within their hearing range, start looking around/ move towards source of sound
* If enemies in a cluster (multiple enemies with overlapping hearing range) and one of enemy is killed, all enemies in cluster will become alerted
* If one of enemy in cluster killed and was targeting player, all enemies in cluster will target player

#### basic enemy

base health: 1 headshot/ 1 regular shot + 1 knife (30 hp)

Dmg: 10 dmg

Attack speed: 2 sec

weak point: head

Movement speed: slightly slower than player (45, according to players placeholder value)

Attack pattern: enemy will get near player before attacking. If enemy is very close to player, it will lunge forward slightly and bite. If player is within max attack distance of enemy, enemy will swipe their arm at player

|  |
| --- |
| Attempt to make enemy dying condition |
| Ignore the event hit blocks, i made it after deleting my attempt to pass in projectile reference |
| * Tried to make a variable that takes in any actors, but then realised i didnt know how to pass projectile reference from its own class to entity taking damage. * Planned to have enemy check for |

|  |
| --- |
| Attempting to reduce health from enemy |
|  |
| Problem: code stops executing at cast to weapon class, but doesnt display cast failed message   * Casting to weapon class to get weapon damage * Need to find alternative way to accessing weapon class and getting weapon damage OR bullet inherit weapon damage the moment it is spawned in when player shoot |
| Accessing weapon class |
| * Will be very slightly more resource friendly(?)   > projectile wont need to have dedicated variable for storing damage  > can be expanded to other projectile but not ammo types like grenades   * Might be more prone to error if cant refer to weapon   Possible solution: hit by bullet type, try to get weapon player currently holding |
| Getting weapon reference through player, then getting weapon damage to minus off from enemy health |

How to run an event only once (events happen once enemy dead):

Refer to data asset documentation

[Data Assets in Unreal Engine | Unreal Engine 5.4 Documentation | Epic Developer Community (epicgames.com)](https://dev.epicgames.com/documentation/en-us/unreal-engine/data-assets-in-unreal-engine)

[forums.unrealengine.com/t/how-do-you-work-with-one-time-events-that-happen-once-in-the-whole-game/642719/6](http://forums.unrealengine.com/t/how-do-you-work-with-one-time-events-that-happen-once-in-the-whole-game/642719/6)

<https://forums.unrealengine.com/t/why-use-data-assets/726093/4>

I a bit dont dare do this since i dont really understand the documentation (scrapped and to the memory bin :D)

### Advanced enemy

Description: buffer version of regular enemy (also has bigger hit box) but with a special move (not decided yet)

Base health: 3 headshots/ 4 regular shots + 1 knife (90 hp)

Dmg: 25 dmg

Attack speed: 5 sec

Weak point: stomach?

Movement speed: slower than regular enemy (30)

Attack pattern: it does a smack. If enemy is far away from player, it will try to charge towards player (lol this seems too much, sounds kinda op)

Other notes: this enemy should be accompanied by at least 2 other regular enemies so player have challenge of ammo management and taking risk

### Enemy categories

Basically just enemy appearance

**Types**:  
 - civilians: will be more likely to drop ammo and med kit

- scientists: not decided yet kek, just according to lore, scientists in lab would be less likely to be prepared for post apocalypse

## Entity

### Entity manager

Handles updates for all entities, player included. Will be separated into children for different tasks.

----------------------beneath this point is dumb me ranting about unreal--------------------

for fuck sake, unreal has a friend tick box but not have include or an actual friend relationship between blueprints smh. so this class will be made as blueprint interface so data can be shared among each other. Actually, idk what im doing hahahahahha. Apparently, this is what a interface does – “A **Blueprint Interface** is a collection of one or more functions - name only, no implementation - that can be added to other Blueprints.”. I swear to god, the fuck does name only no implementation mean O|<.

Ughhhhhh >:v

I guess I’ll just stick to using blueprint and make the classes be a child of entity that is child of actor. Indirect actor inheritance yay O|<

Apparently according to a [stack overflow thread](https://stackoverflow.com/questions/12854778/abstract-class-vs-interface-in-c), interface means a class with pure virtual functions. What’s pure virtual again :’D

Software engineering is fun :’)

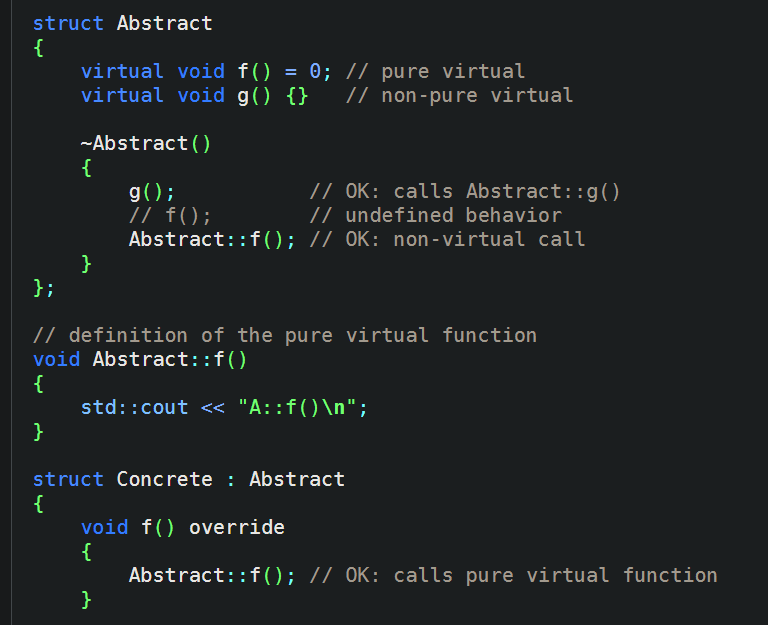
So virtual functions are functions that can be overwritten by child, but a pure virtual function is a function that is not defined in base class but defined in a subsequent child class. also, cannot make an object from said function, though in this case, calling pure virtual as abstract will be clearer. It does not make sense to make an object from an undefined class. turns out im just dumb and need things to be explained like im 5 haha :’D

----------------------------------ignore dumb ranting above-----------------------------------

Trying to picture all objects in an entity manager…

Entity manager is a child of entity, which will be used to store references to all objects in the game AND call for their update. Idk man it sounds weird in the head that you can store a root concept that grown into an actual item.

Note to self: turns out, its possible to define and call a pure virtual function as shown below. I really need to look at how singleton code work again O|<



From [Abstract class - cppreference.com](https://en.cppreference.com/w/cpp/language/abstract_class)

[Singleton pattern in c++ - Blueprint - Epic Developer Community Forums (unrealengine.com)](https://forums.unrealengine.com/t/singleton-pattern-in-c/430452)

^ thread on singleton but in visual studio. Guess im making a cpp file.

Never fucking mind, the thread be talking about issues MAKING the singleton, and guy really said one solution be to not use a singleton :::::::DDDDDD

This is a pain. Im just gonna make this interface class to act as entity abstract class and have a child class to store all the entity references. in other words, entity will be a pure virtual class that acts as a parent for almost every other class, while the direct children to pure virtual entity class will be abstract for the grandchildren classes (different types from parent abstract object).

All entities will be added to entity manager for thread updating, so game manager will only need to call entity manager instead of calling other object class functions themselves.

New updates, yayyyyyyyyyyyyy

Because I have a feeling whatever rambling I typed above wont work (idk, I write so much even though I get put off by long text O|<). Anyways, I just remembered what its name is called: linked list :D

A diagram of a circular linked list

Description automatically generated

Anyways, problem always lies in translating regular c++ code into blueprints

ugh (三 ‘ –‘)

unreal engine doesn’t have vector list, and apparently, they recommend using TArray lists. Then we got the problem of unravelling whatever the mess of ideas I just dump somewhere in this chunk of words. (random thoughts: how many times have I referred to my work as dumb now, idk but I know its true~ In other news, I think my sanity is snapping soon :D) This time, I got some better references on how to use interfaces now :D. Honestly, there isn’t really a need for a interface since I went all the way to make an empty entity blueprint class just for the sake of having everything be shoved under the same parent class, but I guess I see how things goes??? Also, I made an interface communication between enemy and ally, though im not sure if I want to delete and recreate, or just swap out actor components. Yes this is written after the whole re-planning of actor composition.

### Class changer

To change enemy entities to ally entities. Current plan is to make an ally class entity and send enemy info to it before deleting said enemy. This will be a branching “Factory” from entity manager class. enemy reference will be passed into this factory to be copied and spawned in as an ally class object before deleting initial enemy. Factory will then return new ally object and replace initial enemy’s spot in the entity list.

Do I really need to separate this as its own “factory”? making it a child means there will be an excess list created with no functionality. Then again, im not proficient in software engineering O|<

Apparently there’s something called adapter design pattern. Pretty much what this class need to do.

This design pattern needs experience to pull off haha. BUT, more scuffed plans because brain trying to gain a wrinkle. What if the class changer simply maintains a link between ally class to enemy class? since adapter pattern is meant for maximizing reuse and flexibility? Instead of hard coding the ally class to have pretty much the same behaviour and attacks as it once had as an enemy, why not have the class changer call the enemy functions and edit its behaviour to support the player instead.

Player will use resurrect on dead enemy. Entity manager needs to know that to-be resurrected enemy is undergoing class change, so its class pointer needs to keep reference to this actor.

Quick search into changing class of instantiated object later, I has found existence of strategy pattern. since it is more for algorithm, it will be used for enemy behaviour. But enemy will eventually become ally, and ally needs to adapt to friendly behaviour. Extremely scuffed, but edit strategy pattern with adapter pattern? ‘ –‘

Sounds like a headache ahahahha

A diagram of a diagram

Description automatically generated with medium confidence

P a i n

As you can see from diagram above, changing an actor’s class involves changing their behaviour (if it exists). Honestly in my head, it’s just the case of modifying the target to something else. But who knows what errors the system like to throw amiriteeee hahahhahahah. By the way, who knows if these patterns would work in unreal engine, the software that has issues handling singletons ‘ –‘

My bad, I misunderstood the bit about client interface. I mean, I knew it’s meant to be more like a view only class, but damn im dumb. BUT THE BOOK SWOOPS IN TO SAVE ME AND MY DUMB ADAPTER STRATEGY PATTERN.

Current problems so far (old ver, before this whole paragraph of small braining):

* Having issues transferring enemy components to ally entity. This version of code does not have class changer class, but the problem still remains. Need to figure out how to detach all components and attach to new target.
* Reference to enemy types. Currently everything is handled by enemy base class, children enemy types only contain differing models to separate from each other (will also contain “special” moves). Children are currently referring to parent base class to handle every event. Need to figure how to get out of parent back to class that called base class to get info of its components (to change class).

Ally vs enemy

ally is basically enemy but friendly to player. So instead of targeting player, all that needs to be changed from enemy class is the target. But ahahaha, I just had to make a while damn mess of a system for myself for no good reason. This is just a short paragraph meant to identify the difference between the 2 classes (so I can minimize the headache I get while setting up adapter pattern)

**MORE CHANGES TO PLANS BECAUSE I AM DUMB TO NOT RESEARCH BEFOREHAND**

So, apparently its not possible to change classes of an object after said object been spawned in :D

BUT, we still have plan to copy components over new actor before discarding old enemy actor (by components, I mean their physical appearance component, so they appear the same. I am being a terrible software engineer because this is a complete mess :D).

Either ways, this new plan sounds more like adapter pattern now. Honestly, I got this idea from a reddit thread answering poster’s question of if it’s possible to change classes. Replies said to either use tags to change “class” or replace the components that make them identify themselves as their role (essentially the entire point of adapter pattern, not whatever scuffed bullshit I tried to pull off above).

In summary, these are the changes im going to make:

1. Instead of making a class containing all the objects’ appearance and behaviour, make separate the class definition into smaller components that can be attached to an object. This allows the object to be able to swap part of its components to receive “updates” to its changes.
2. An actor (or object in scene) will be made of mainly these components: appearance and behaviour states.
3. The behaviour state component will continue to use strategy pattern as planned. This will allow the algorithm to decide its behaviour instead of manually coding them into separate children type classes. Saves more storage space yay. Sounds like a pain to search through and debug though haha :’D

Algorithms banzaii O|<

That leaves with how to tackle enemy type change and usage of adapter pattern to change allies’ behaviour O|<

Lets just say, hopefully I can create a working system that allows me to overwrite the attack target during runtime, then I can just do this strategy in adapter class or something.

As for enemy types though (ughhhhhh), maybe I can split the appearance components into more components? I think I’ll probably use strategy pattern to do this too (since its just the change of models). I need to make this super clear so I don’t mess up later hfwuosdoa.

Actor on scene is identified by:

* Appearance component
* Behaviour component

Since appearance and behaviour change according to enemy type, both components will have 3 children classes/ strategies.

A diagram of a person with text

Description automatically generated

how diagram changes when enemy converts to ally:

A diagram of different types of type of type of class

Description automatically generated

All that’s left will be how to identify the actors… Tags. WHICH MEANS I DON’T HAVE TO TYPE LONG NAMES TO GET THE PLAYER REFERENCE YAYYYYYYY

Anyways, I don’t think there will be a need to ever check the specific type of ally/ enemy, so there wont be a sub category for different types.

## Inventory

* stores all resources player collected
* sort by id num
* can be sort by item number (descending, ascending)

##### inventory categories:

* crafting materials
* quest items
* Armoury (idk wat word to use that include bandages and meds but yes they will be in here too)

> put bandages and meds at the front so it's easy to open and use during combat

> shift equipped weapon and weapon bullet to front of list for search

##### 

**Sorting:**

* Sort by amount, alphabetic name using data structure nodes

##### what info each item have:

* id number (int)
* item name (string)
* item amount (int)
* Inventory category (enum)
* item description (string)

**Item highlight:**

To clarify which item will be picked in the case where there is a cluster of pickable items close to each other, an obvious marker will appear around the item. Initially wanted to do an item highlight, but i didnt know how to do that. So i improvised and placed a point light in the item itself. Now when the player steps into the item collider, a pick up prompt will appear and the item will be “highlighted”.

**How to code to picking up of items:**

Currently, all key press events are handled in firstPersonCharacter blueprint (class). Picking up of items is to be handled in item base class (though now that i think about it, maybe i really should have made item types as children classes instead of enums or tags O|<. on the bright side, just started on it so i can technically still make the changes ‘ -’). With current intention of items to be picked up by player when player and item sphere overlap, game needs to check if player pressed interact/ pick up button. Item will also need to checked for “pickable” tag.

Conditions to check:

{ check “pickable” tag exist on item + interact/ pick up key pressed + player within item collider range }

Since picking up will only be doable when player within item collider range check the rest of condition within collider overlap event

Existing code:

{ item collider overlap event| if is player overlapping with collider = item highlighted + show prompt }

As you can the above code, it will be quite messy to expand within this event (its already confusing to think how to organise this). Therefore, instead of checking for overlap of collider with item, player will check if item is highlighted or “pick up” prompt is active on screen

Finalized code:

{ itemHighlighter / “pick up” prompt is active -> check for interact button pressed + item has “pickable” tag -> pick up function { add item into inventory + remove item from scene } }

Problems with separating item behaviour by child classes:

* Dont think its possible to be a child of both classes (yes, there will be some items that is pickable and stackable. And yes, pickable and stackable are the only tags that exists for now).

**Fixing collider and highlighter reference to base class:**

Problem: error referring to point light and sphere collider

Theory: when child class enters parent class to access collider overlap event, parent components

referred to and attempted to access instead of child components. Child obviously has no access to parent component, so throw reference error.

Possible fix:

1. Modify collider overlap event to take in components to be modified. Overlap event will still be handled in parent class, but function should have right reference to child components
2. Parent class still handles overlap collider event, but instead of passing in parent components, code needs to find components themselves then modify visibility. Expected to cause errors because no specify if find components in parent or child. (lower priority fix)

Working fix method:

Error was thrown because while child class calls parent tick function, it did not call begin play function. Parent begin play function is in charge of making “pick up” prompt. Since parent begin play was never called, prompt was never made, therefore causing reference error. Honestly have no idea how itemHighlighter got fixed though. It fixed itself just by making sure prompt message was made.

**Jokes on me, i need to rework the item class framework to pass in child/ own class highlight**

Nevermind i should have checked before malding. Child point light lights up properly (tested with different colours)

## weapons

pistol: 20 dmg, can hold 7 bullets at a time

Knife: 10 dmg

#### [Breaking down of gun class]

Gun components:

* Gun type/ name
* gun dmg
* Bullet

> Should delete itself when it hits any surface

> Deals dmg to anything with health (for now)

|  |
| --- |
| A pitiful attempt to have bullet class handle enemy deletion - 10/6/24 |
|  |
| problem: - bullet class is currently handling deletion of enemy  - bullet class need to check if object hit is from enemy class  - cannot delete bullet unless delete enemy first (because lose reference to delete enemy afterwards when delete bullet first)  - can cause awkward bullet sustain effect (wont really be noticeable, but still) I wasted 2 hrs for this waaaaaooooooow |

* Gun magazine (ammo gun can hold)

> Stop player from shooting if no ammo

* Ammo count (ammo player is currently holding)

## UI

##### Sixth sense:

* Player be moving router and each enemy be a device
* When device is detects router nearby, send ping to router
* Router receives ping from device

(just an example to self about how player receives info about enemy targeting in terms of routers and mobile devices)

* Set ui active

##### inventory:

Rough idea of how inventory will look like

****

Advanced movement

##### sprint:

* Hold shift, increase movement speed

##### vault:

* Player choose to vault
* Set movement bool variable to false so player cant move (for cinematic/ cutscene? vaulting)

> this bool is checked in physics code so player input during this time will not move player

* Do cam tilt to simulate vaulting (no animation since artists not doing character design. They not confident in it, might put in scuffed model from online)
* When cam tilt sequence ends, reset movement bool so player can control player again

Things to fix:

* Movement speed [done]
  + Player moving way too fast
  + Acceleration too drastic
  + Add sprint button
* Tutorial room door
  + Door is always checking for area collision
  + Change to prompt check
* Second enemy encountered in tutorial level
  + Very obvious animation loop replay
  + Maybe change to start moving only when see player
* Boss fight location/ sequence
  + Boss looks very random in empty and open carpark area
* Minimap?
  + A bit confusing navigating around street since theres no level design indicators to guide player to boss
* Medicine shelf at waiting area
  + Medicine bottle labels emitting light (and is very obviously visible)
* Fix tutorial room floor
  + Current floor is made out of multiple tile meshes
  + Figure out how to repeat material texture on a single plane