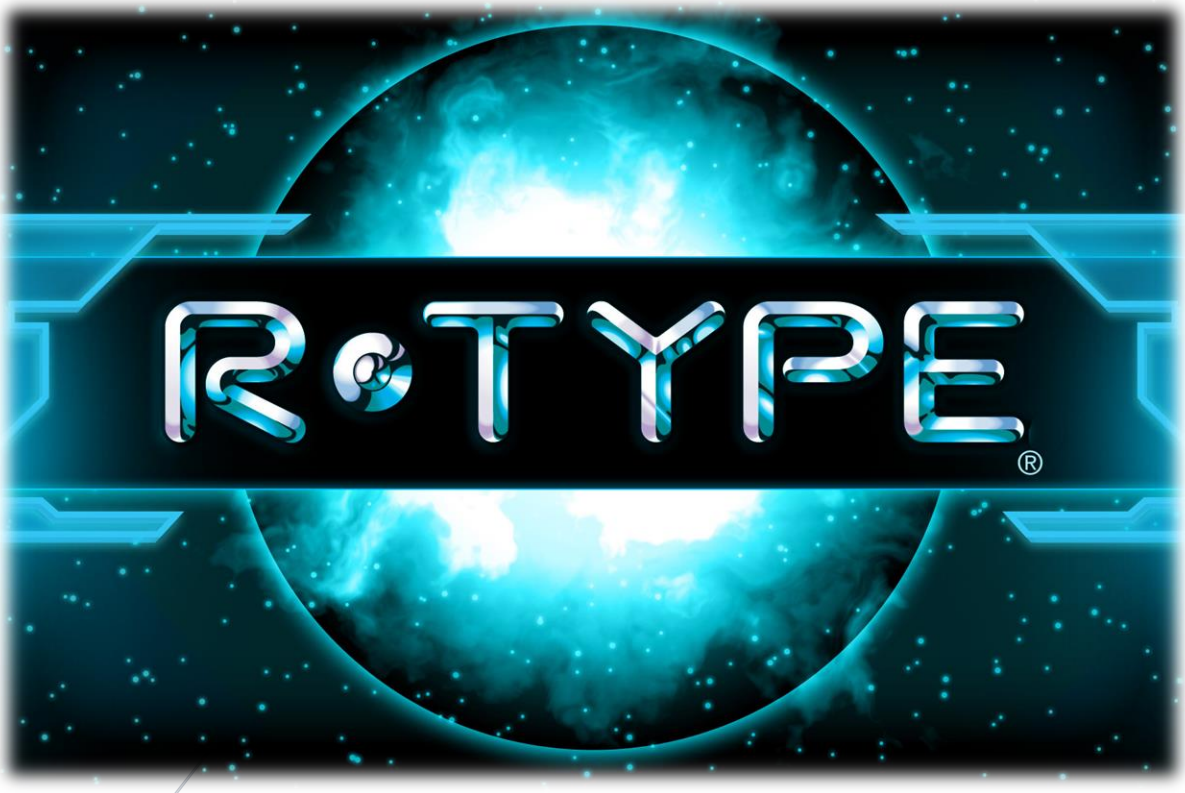
22/11/2016

R-TYPE

Conception



Fernand Veyrier, Quentin Gasparotto, Victor Gouet, Adrien Wery, Nicolas Constanty
SALTY STUDIO

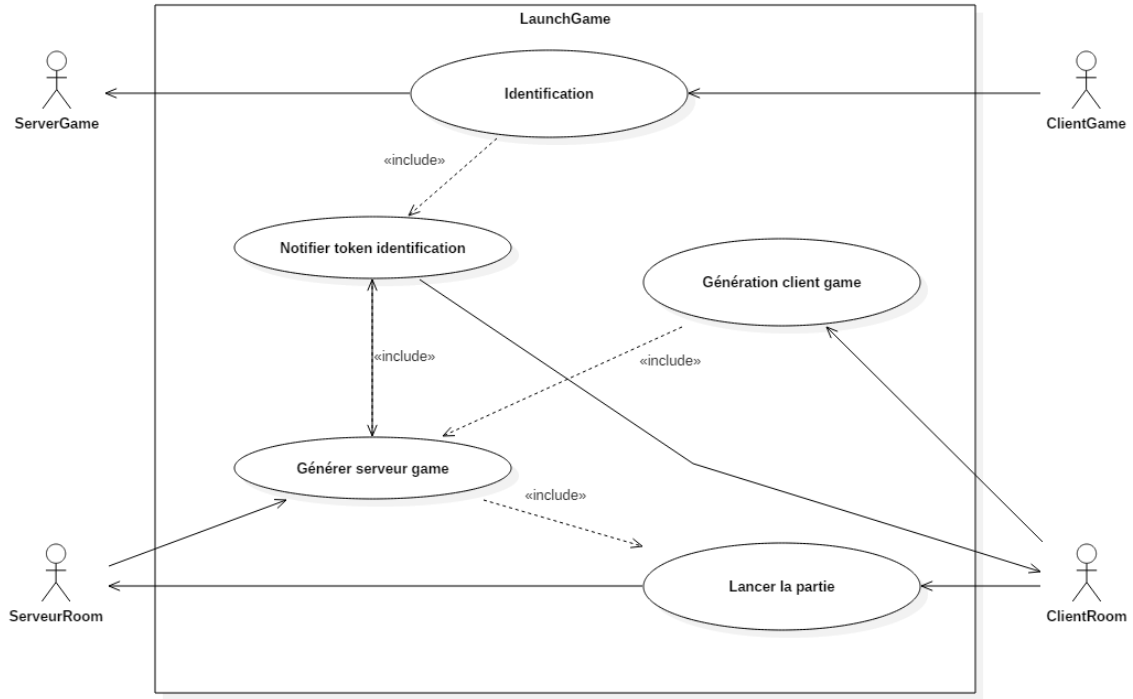
Table des matières

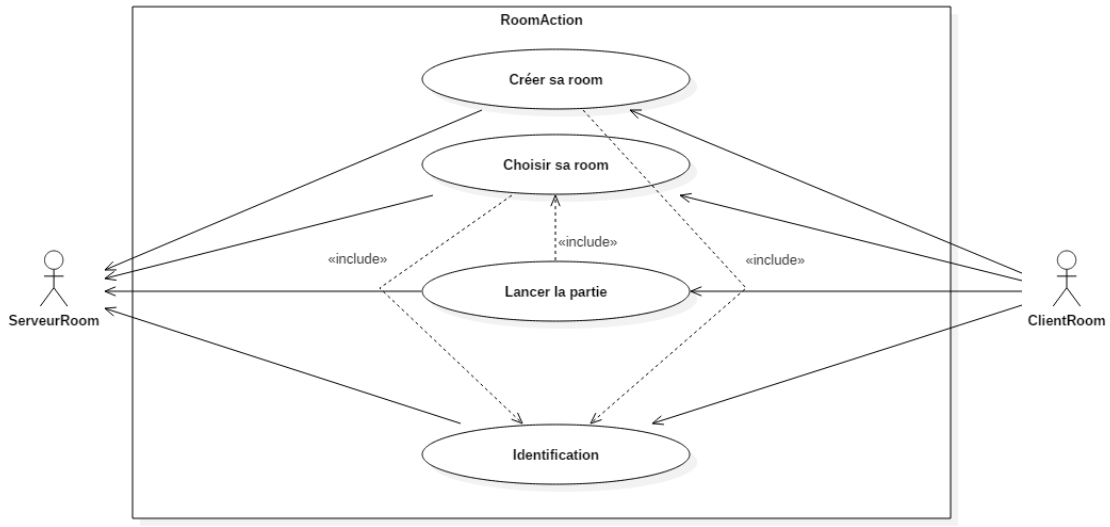
Use Case 2

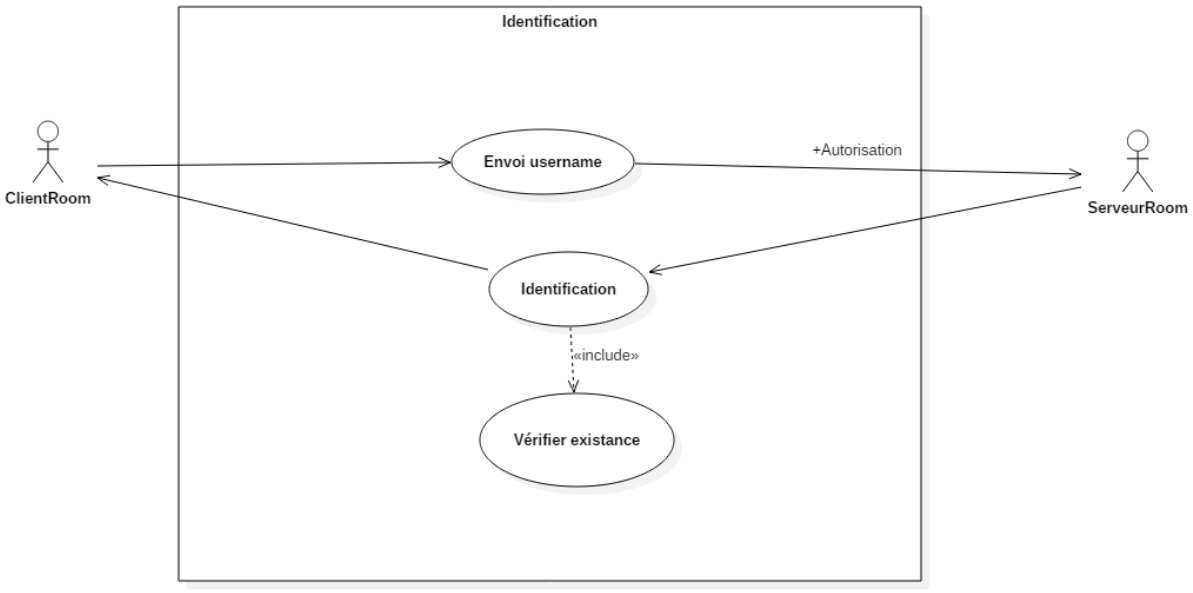
Sequence Diagram..... 8

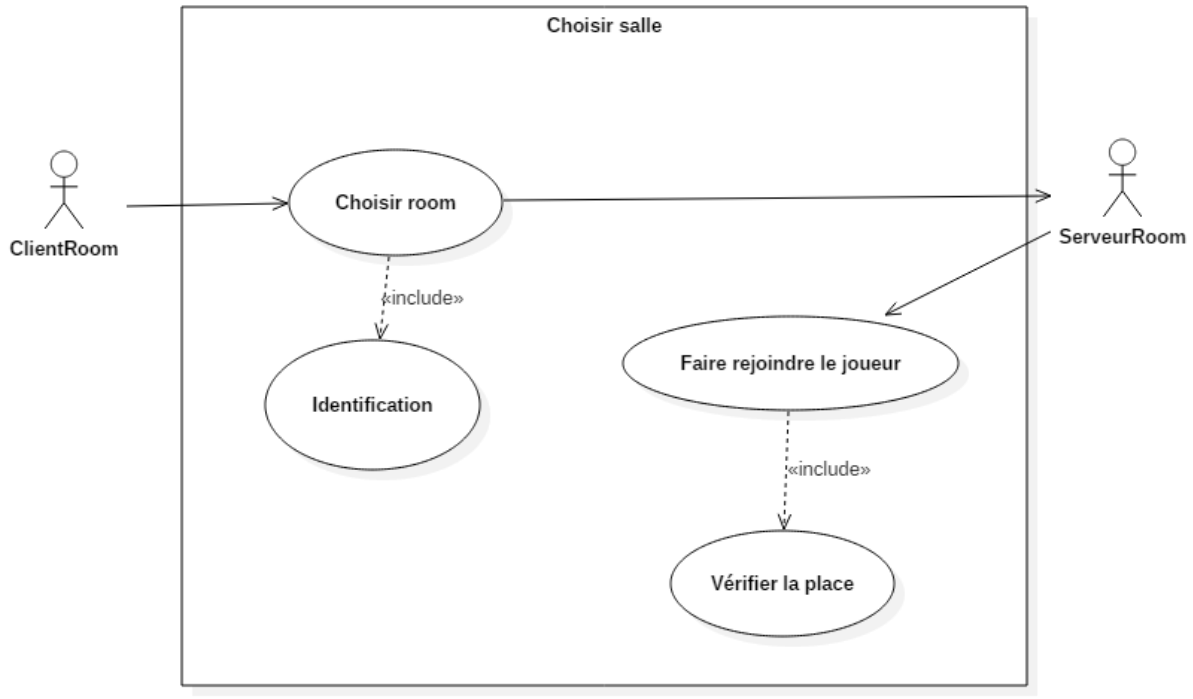
Class Diagrams..... 15

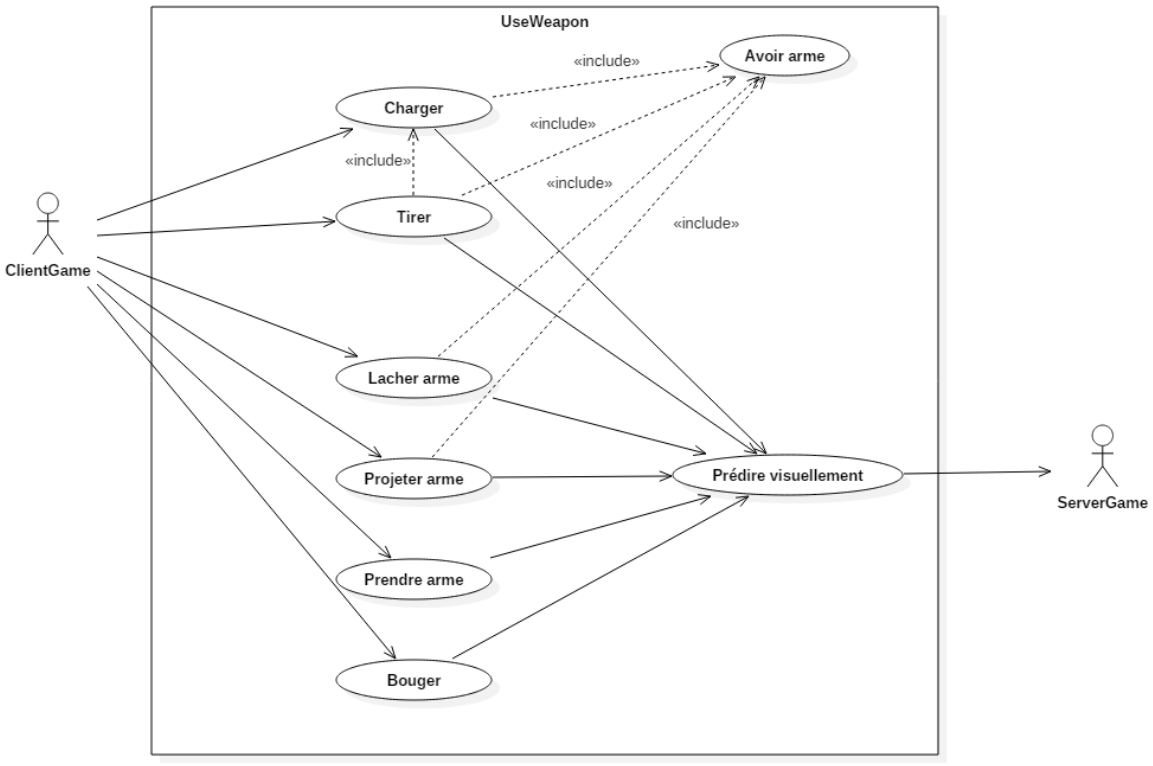
Use Case

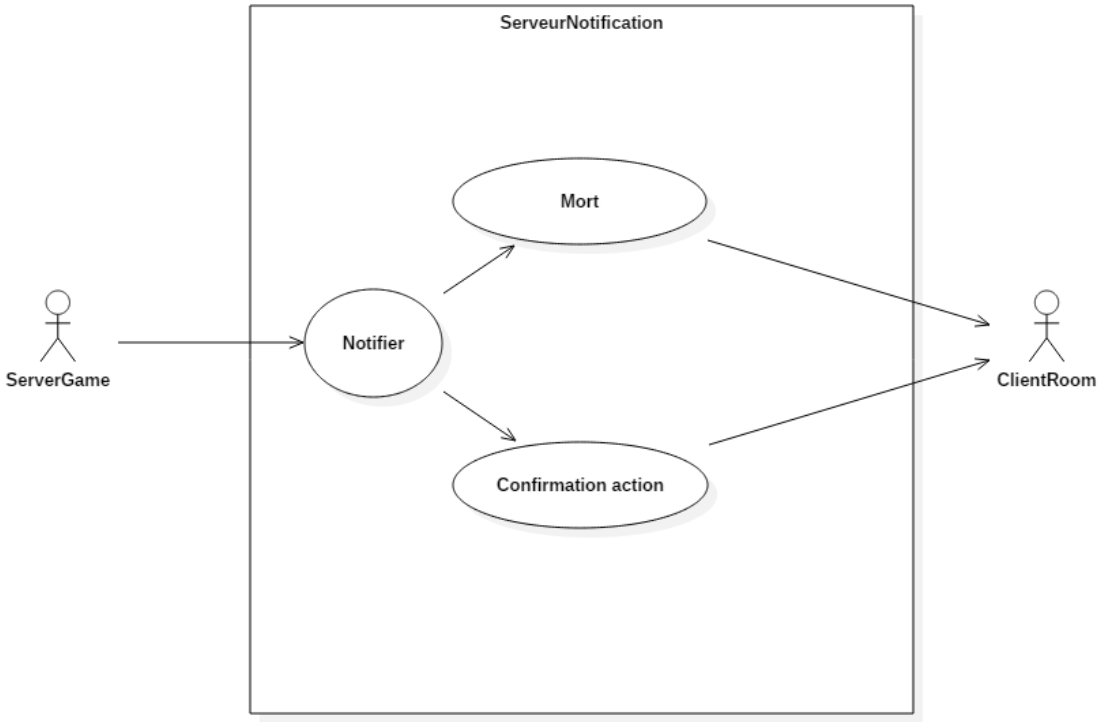

LaunchGame
ServerGame
ClientGame
Identification
«include»
Notifier token identification
Génération client game
«include»
«include»
Générer serveur game
«include»
ServeurRoom
Lancer la partie
ClientRoom

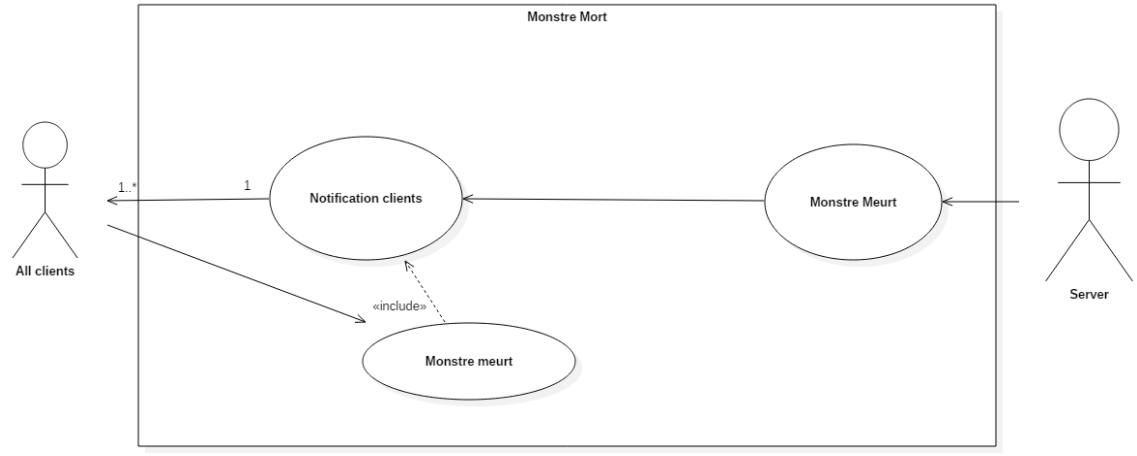

RoomAction
Créer sa room
Choisir sa room
«include»
«include»
«include»
ServeurRoom
Lancer la partie
ClientRoom
Identification

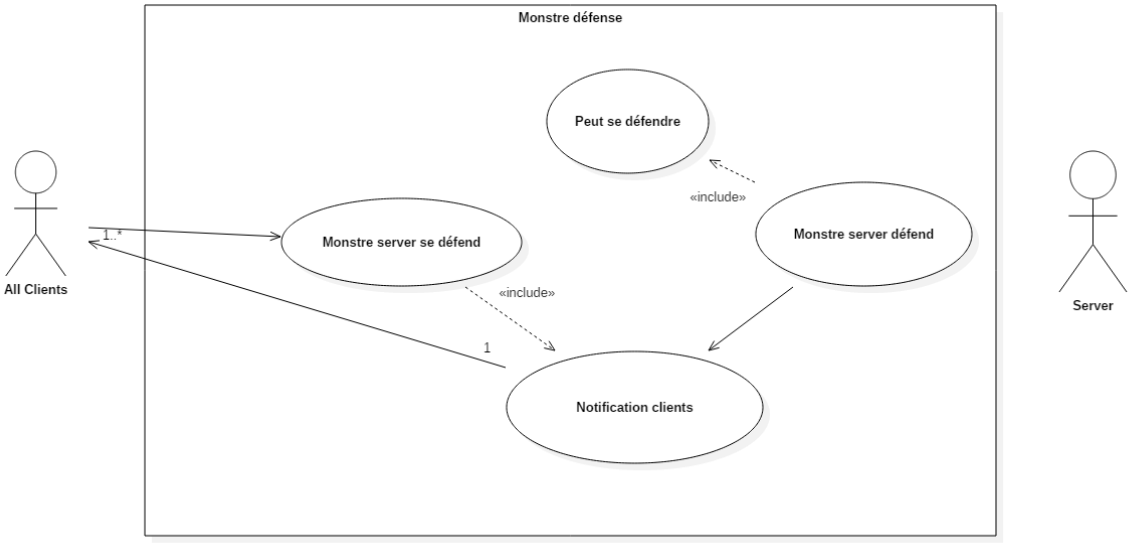

Identification
ClientRoom
Envoi username
+Autorisation
ServeurRoom
Identification
«include»
Vérifier existance

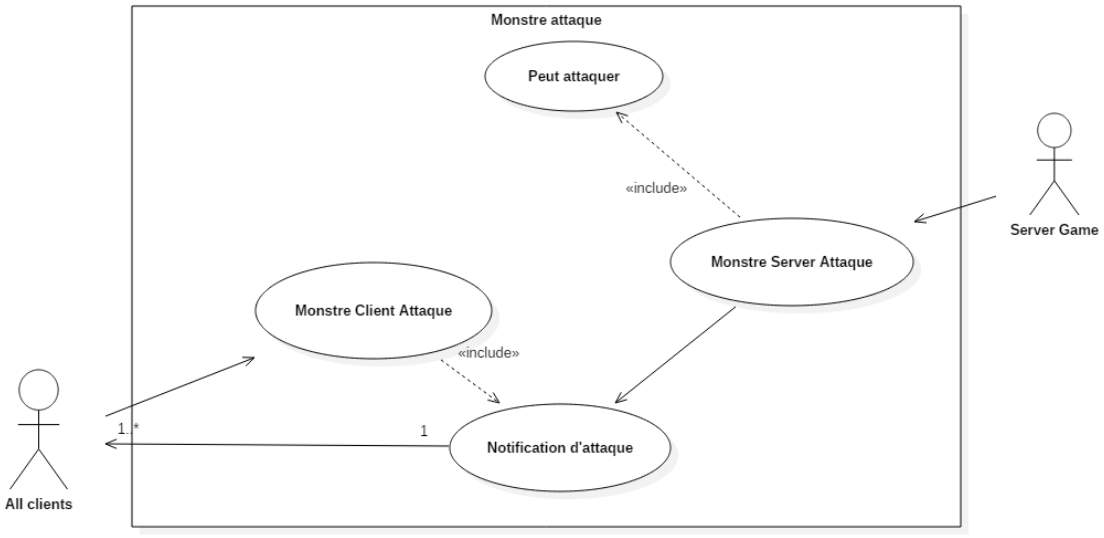

Choisir salle
ClientRoom
Choisir room
ServeurRoom
«include»
Identification
Faire rejoindre le joueur
«include»
Vérifier la place

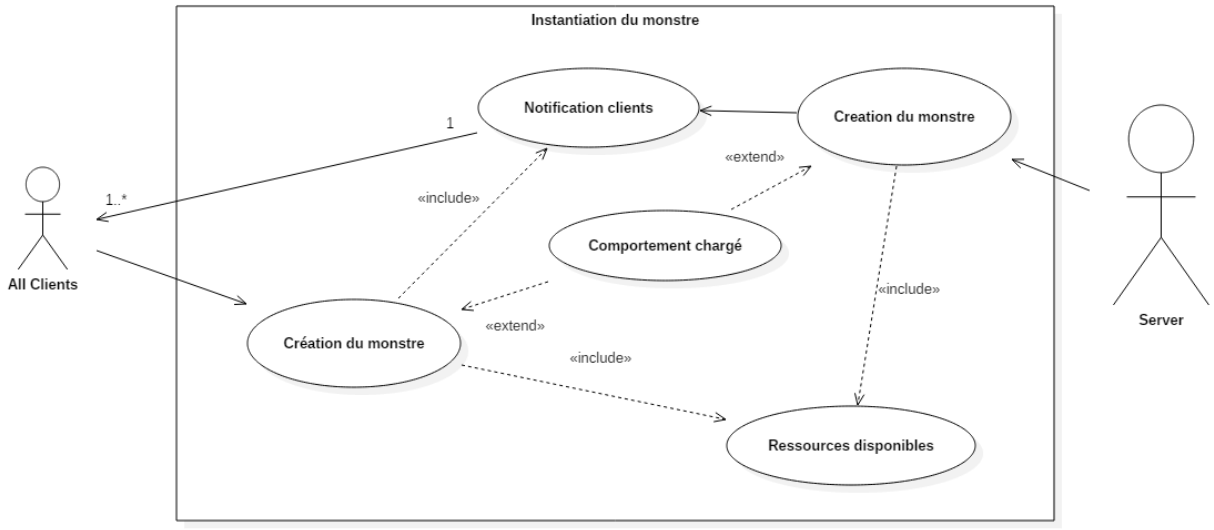

UseWeapon
Avoir arme
«include»
Charger
«include»
«include»
«include»
Tirer
«include»
ClientGame
Lacher arme
Projeter arme
Prédire visuellement
ServerGame
Prendre arme
Bouger

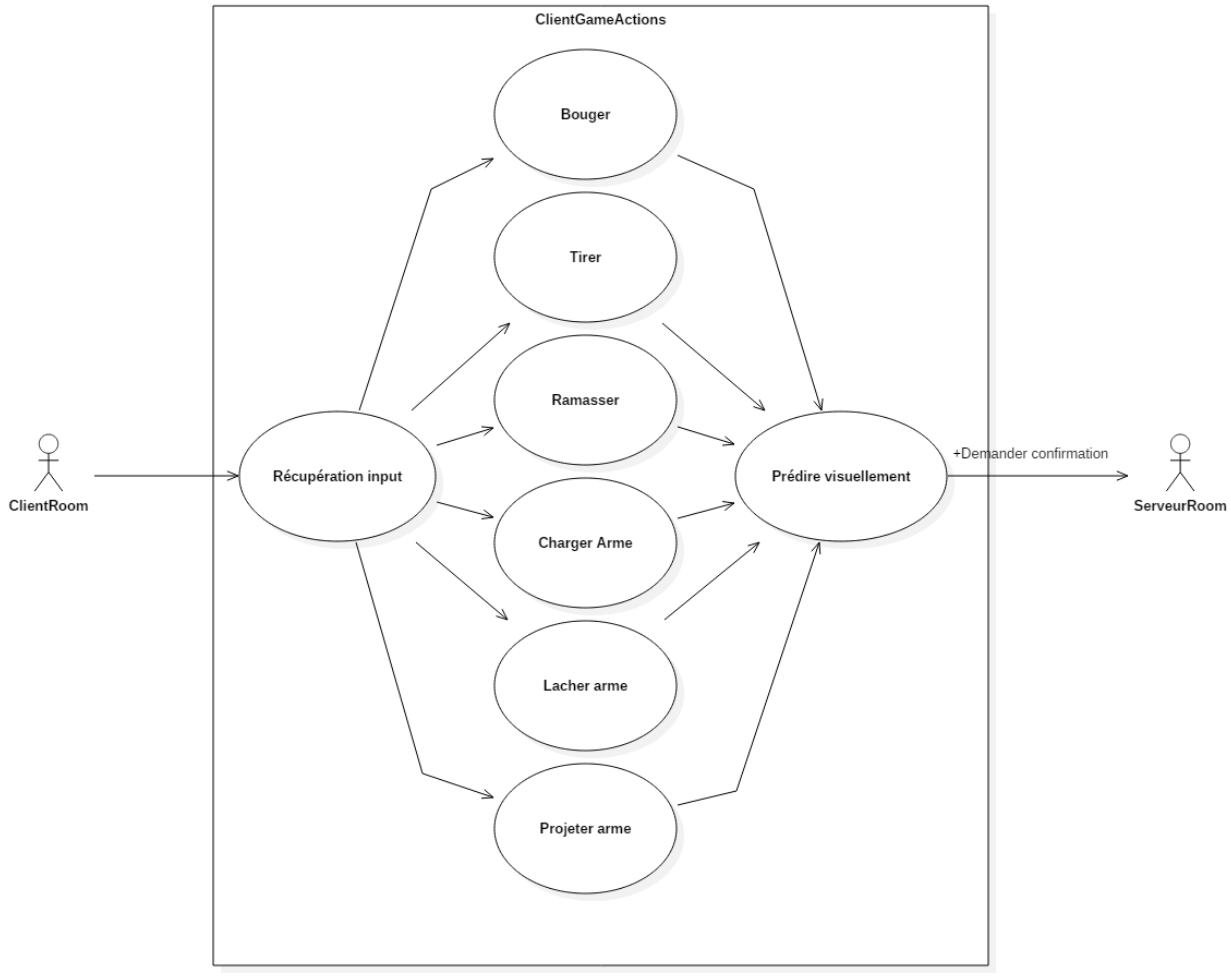

ServeurNotification
Mort
ServerGame
Notifier
ClientRoom
Confirmation action

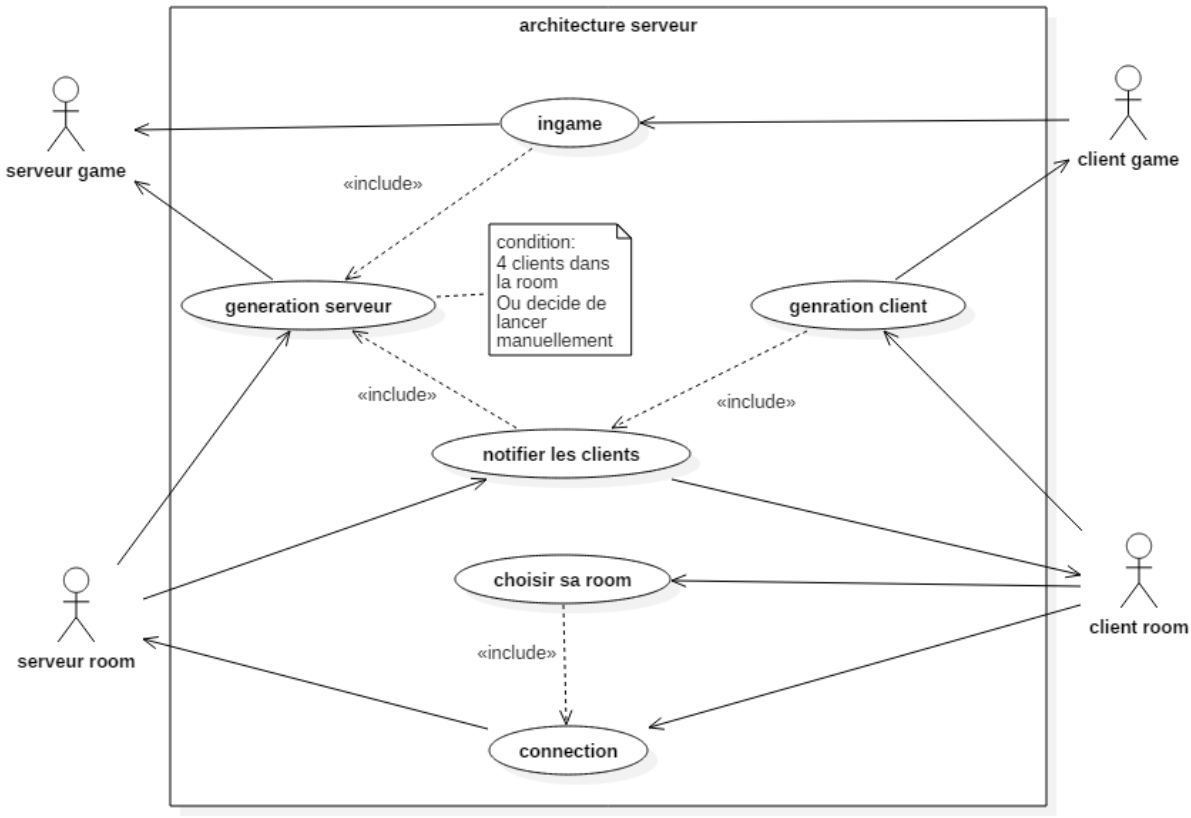

Monstre Mort
All clients
1..*
1
Notification clients
Monstre Meurt
Server
«include»
Monstre meurt


Monstre défense
Peut se défendre
«include»
All Clients
1..*
Monstre server se défend
Monstre server défend
Server
«include»
1
Notification clients

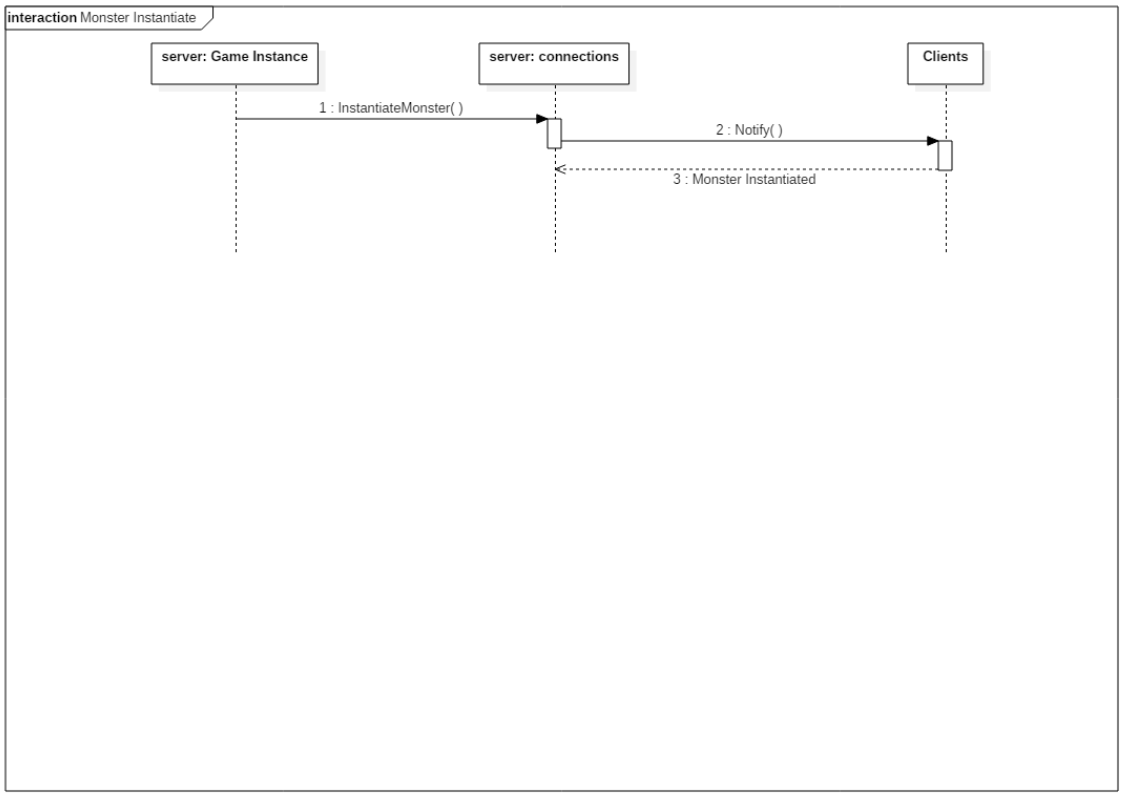

Monstre attaque
Peut attaquer
«include»
Server Game
Monstre Server Attaque
Monstre Client Attaque
«include»
All clients
1..*
1
Notification d'attaque

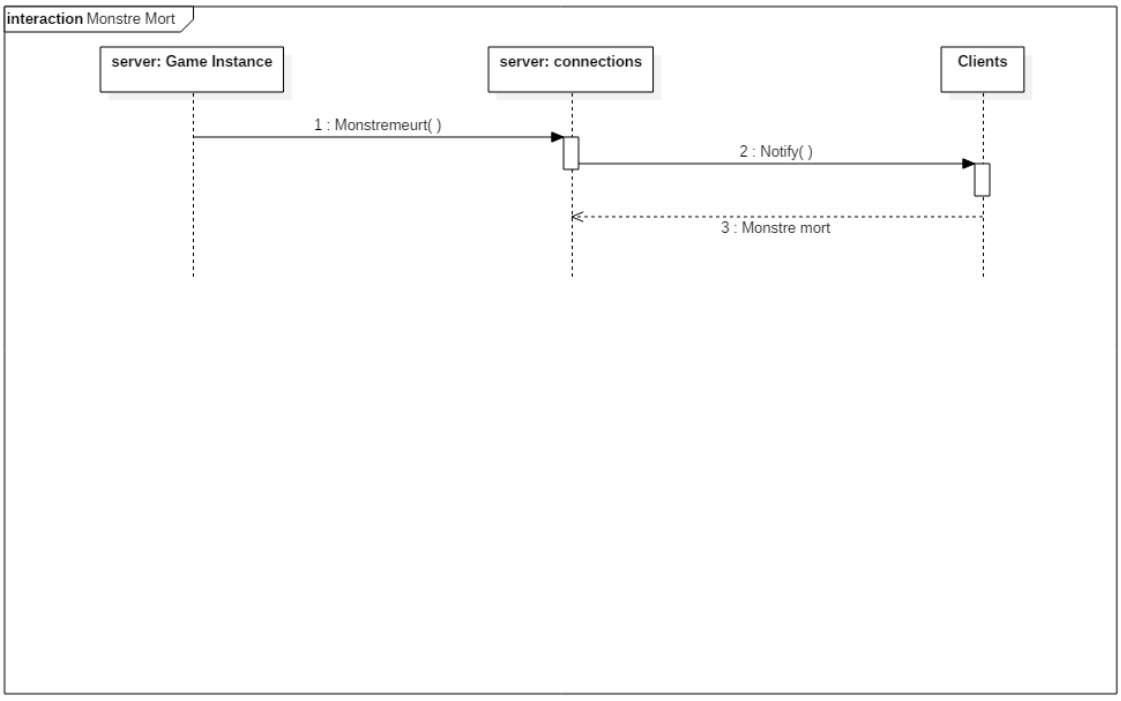

Instantiation du monstre
Notification clients
Creation du monstre
1
1..*
«extend»
«include»
Comportement chargé
«include»
«extend»
Création du monstre
«include»
Ressources disponibles
All Clients
Server

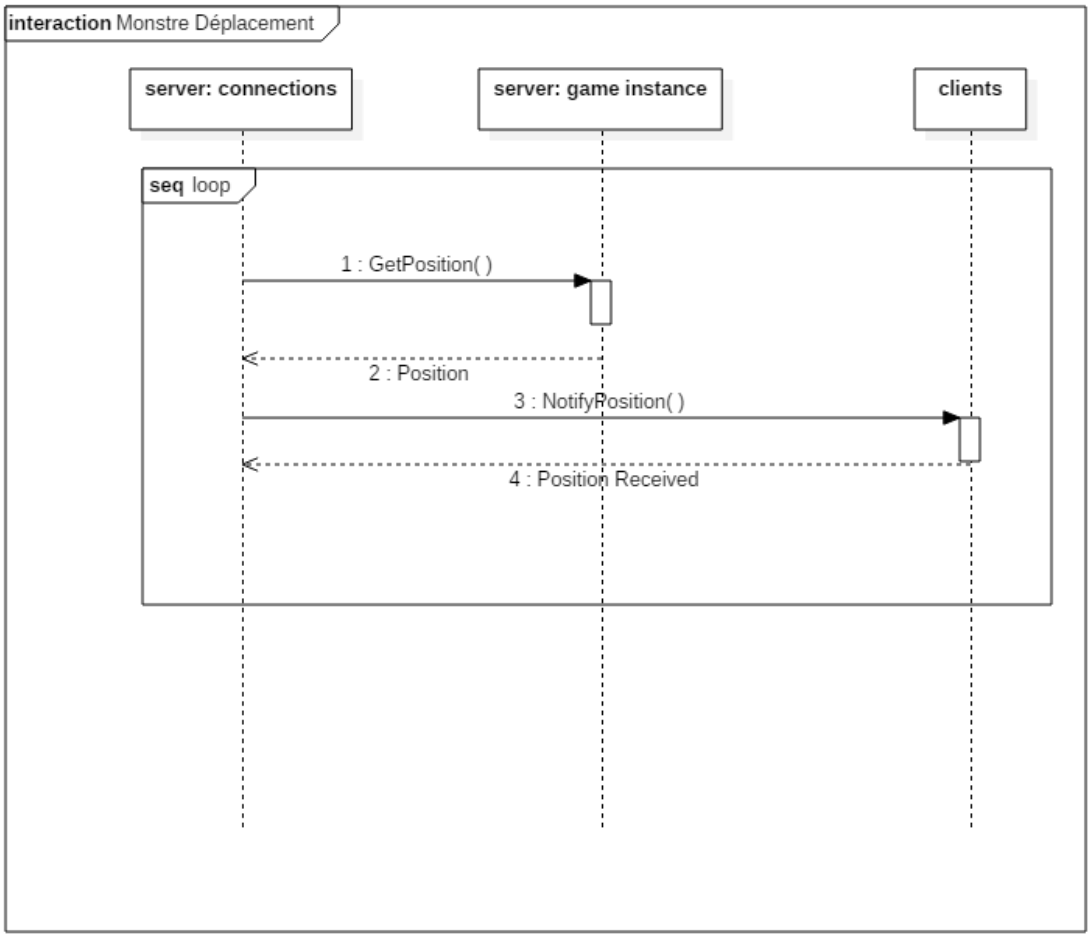

ClientGameActions
Bouger
Tirer
Ramasser
Charger Arme
Lacher arme
Projeter arme
Récupération input
Prédire visuellement
+Demander confirmation
ClientRoom
ServeurRoom

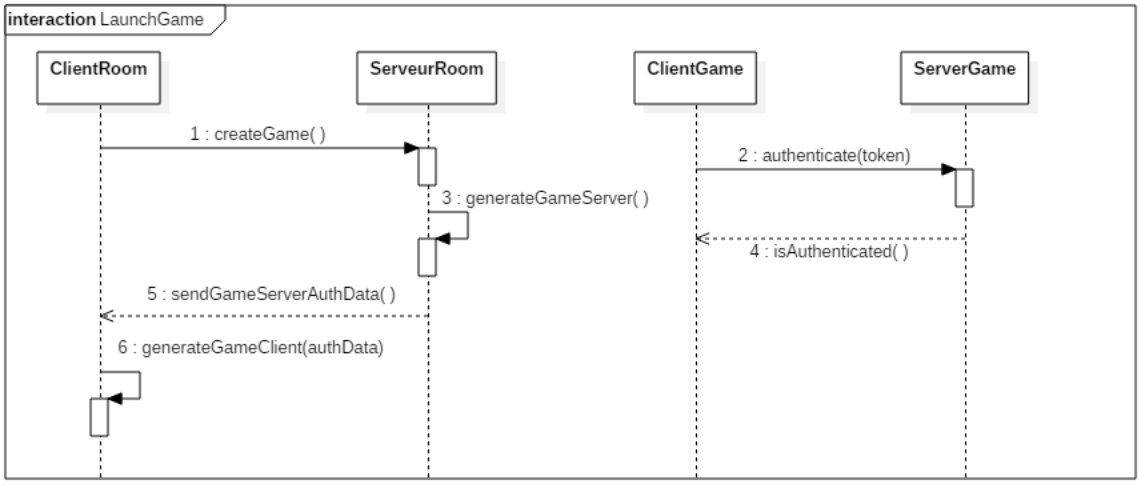

architecture serveur
serveur game
client game
ingame
«include»
condition:
4 clients dans
la room
Ou decide de
lancer
manuellement
generation serveur
genration client
«include»
«include»
notifier les clients
choisir sa room
serveur room
client room
«include»
connection

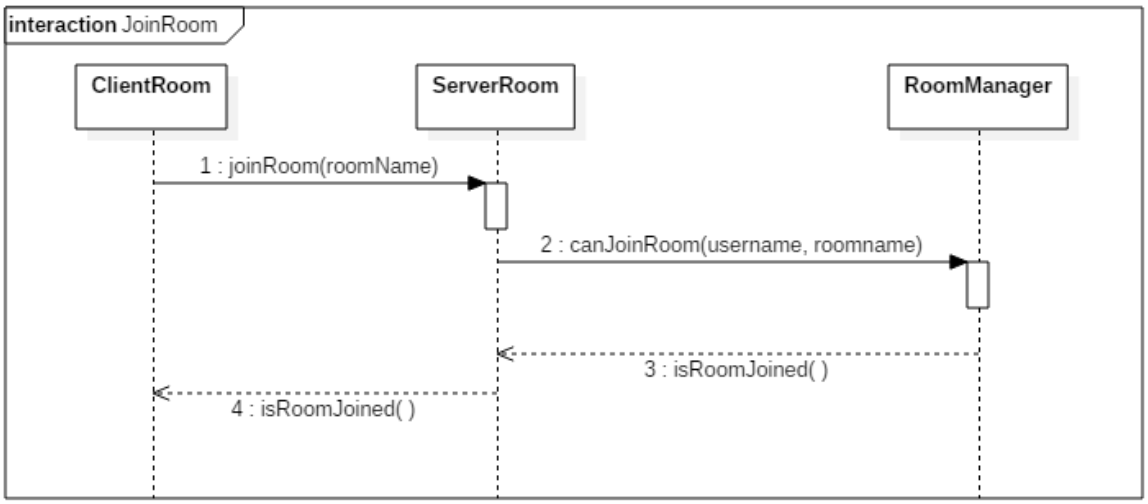
Sequence Diagram

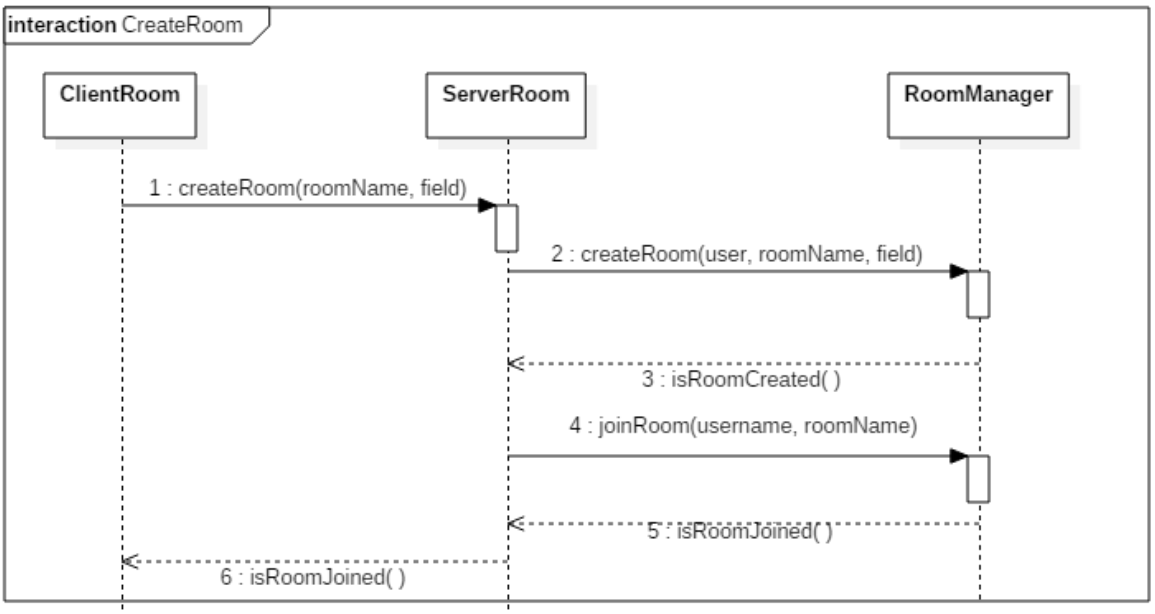


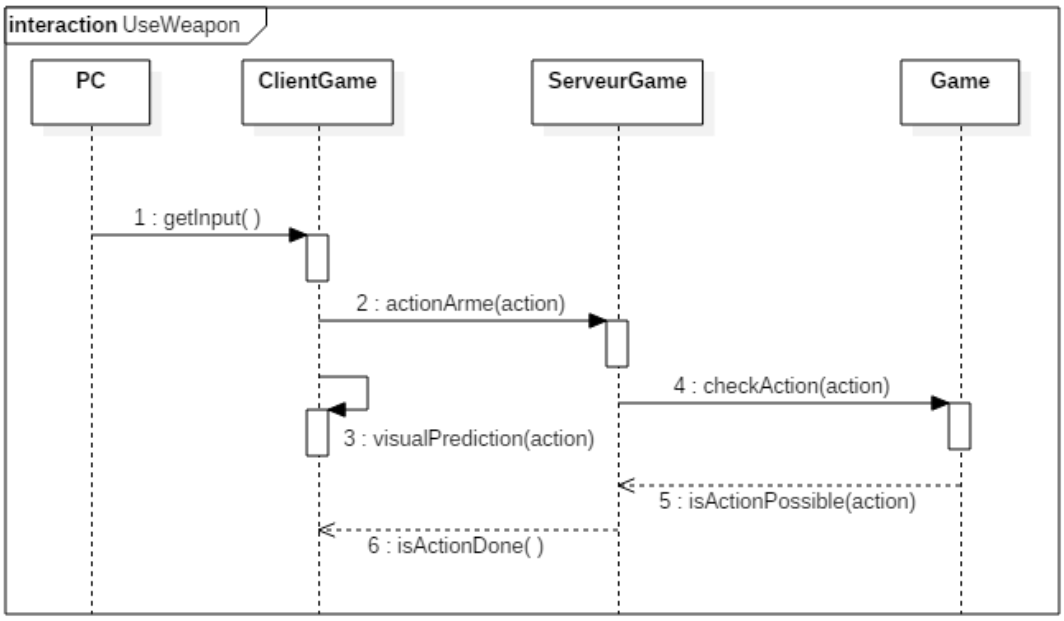


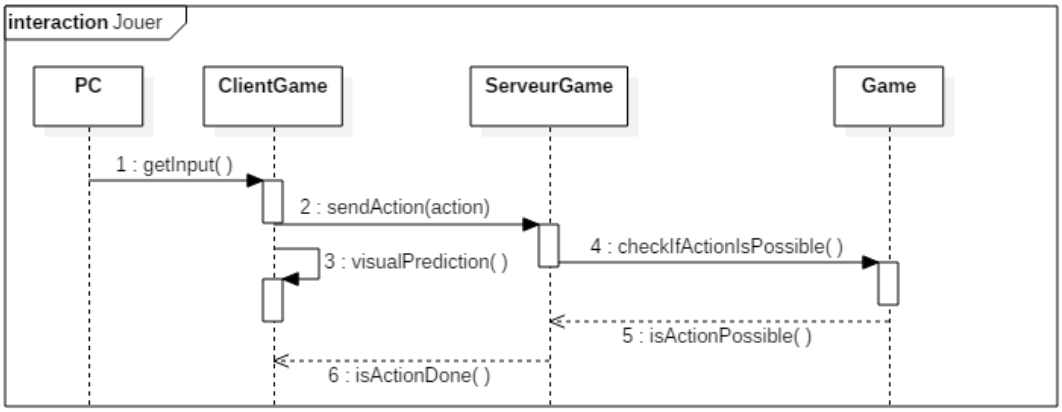

interaction Monstre Déplacement
server: connections
server: game instance
clients
seq loop
1 : GetPosition( )
2 : Position
3 : NotifyPosition( )
4 : Position Received

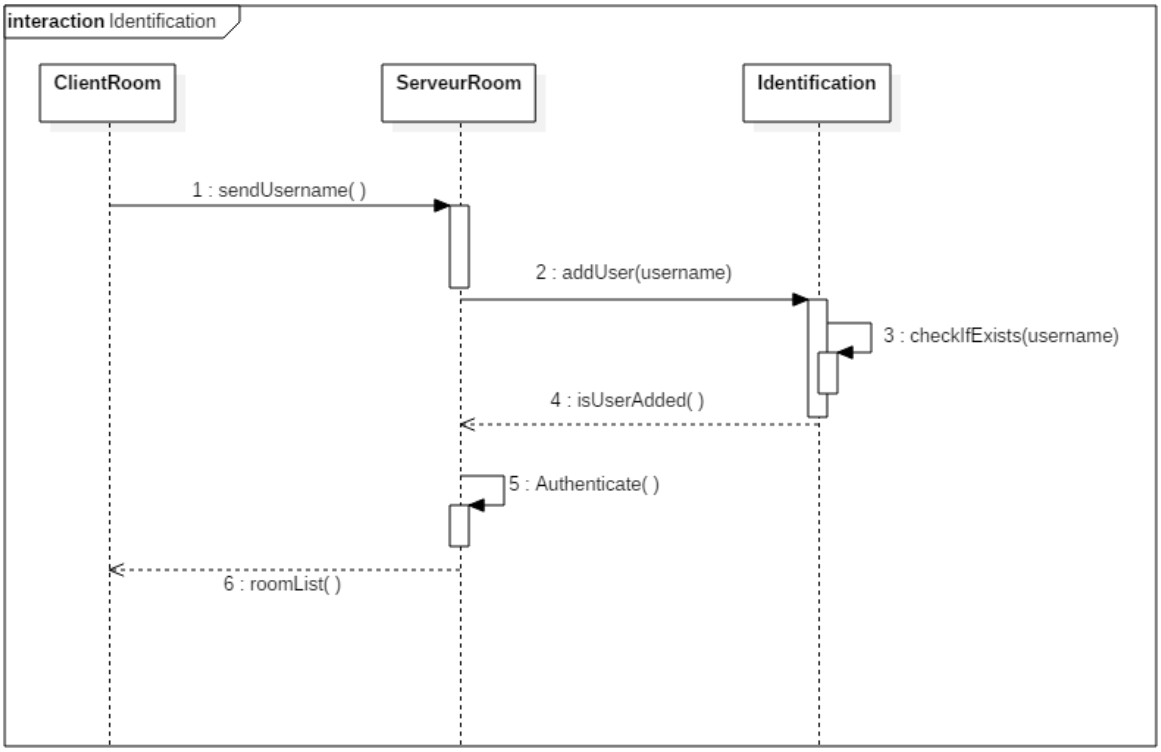

interaction LaunchGame
ClientRoom
ServeurRoom
ClientGame
ServerGame
1 : createGame( )
2 : authenticate(token)
3 : generateGameServer( )
4 : isAuthenticated( )
5 : sendGameServerAuthData( )
6 : generateGameClient(authData)

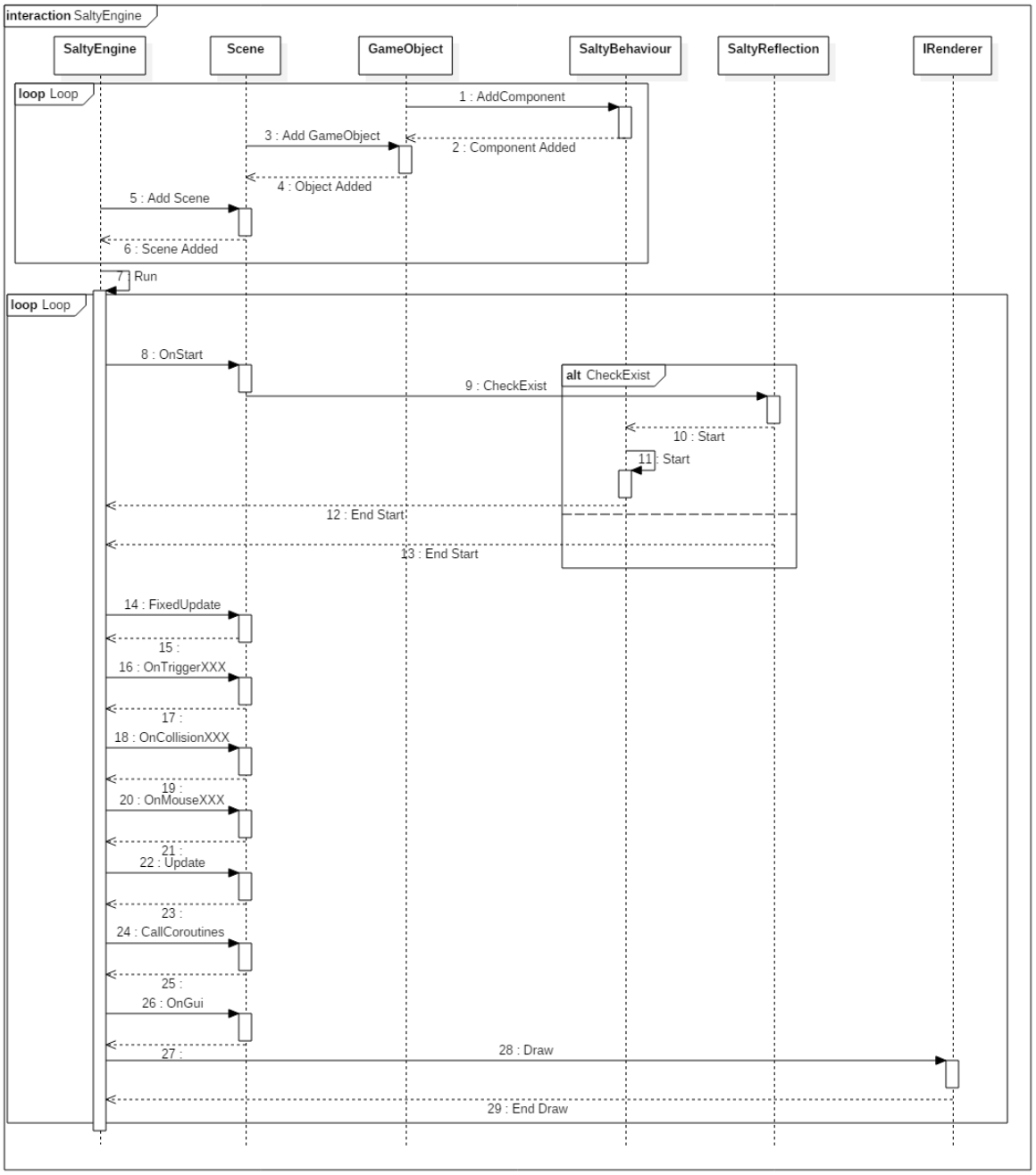

interaction JoinRoom
ClientRoom
ServerRoom
RoomManager
1 : joinRoom(roomName)
2 : canJoinRoom(username, roomname)
3 : isRoomJoined( )
4 : isRoomJoined( )


interaction CreateRoom
ClientRoom
ServerRoom
RoomManager
1 : createRoom(roomName, field)
2 : createRoom(user, roomName, field)
3 : isRoomCreated( )
4 : joinRoom(username, roomName)
5 : isRoomJoined( )
6 : isRoomJoined( )

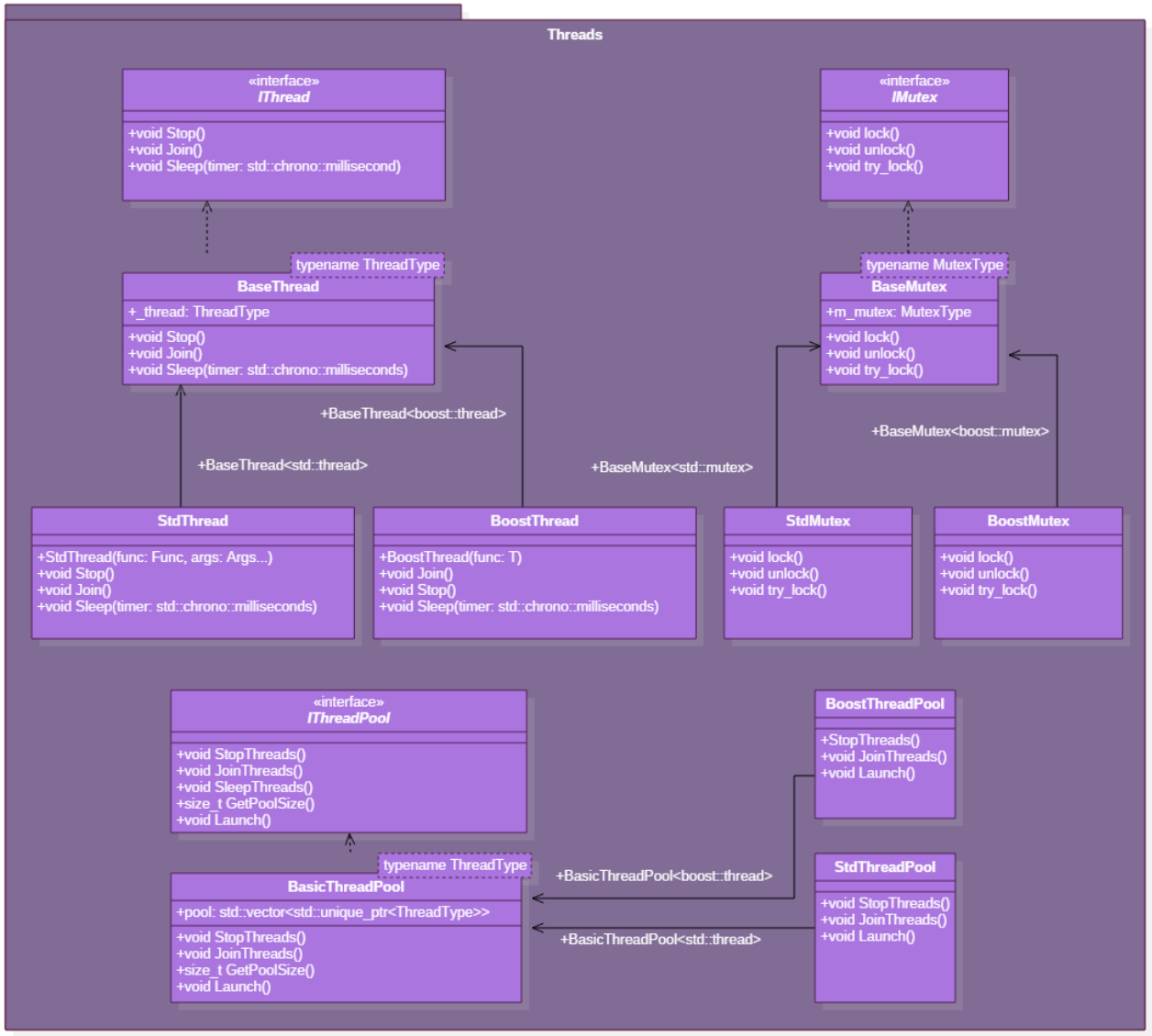

interaction UseWeapon
PC
ClientGame
ServeurGame
Game
1 : getInput( )
2 : actionArme(action)
4 : checkAction(action)
3 : visualPrediction(action)
5 : isActionPossible(action)
6 : isActionDone( )

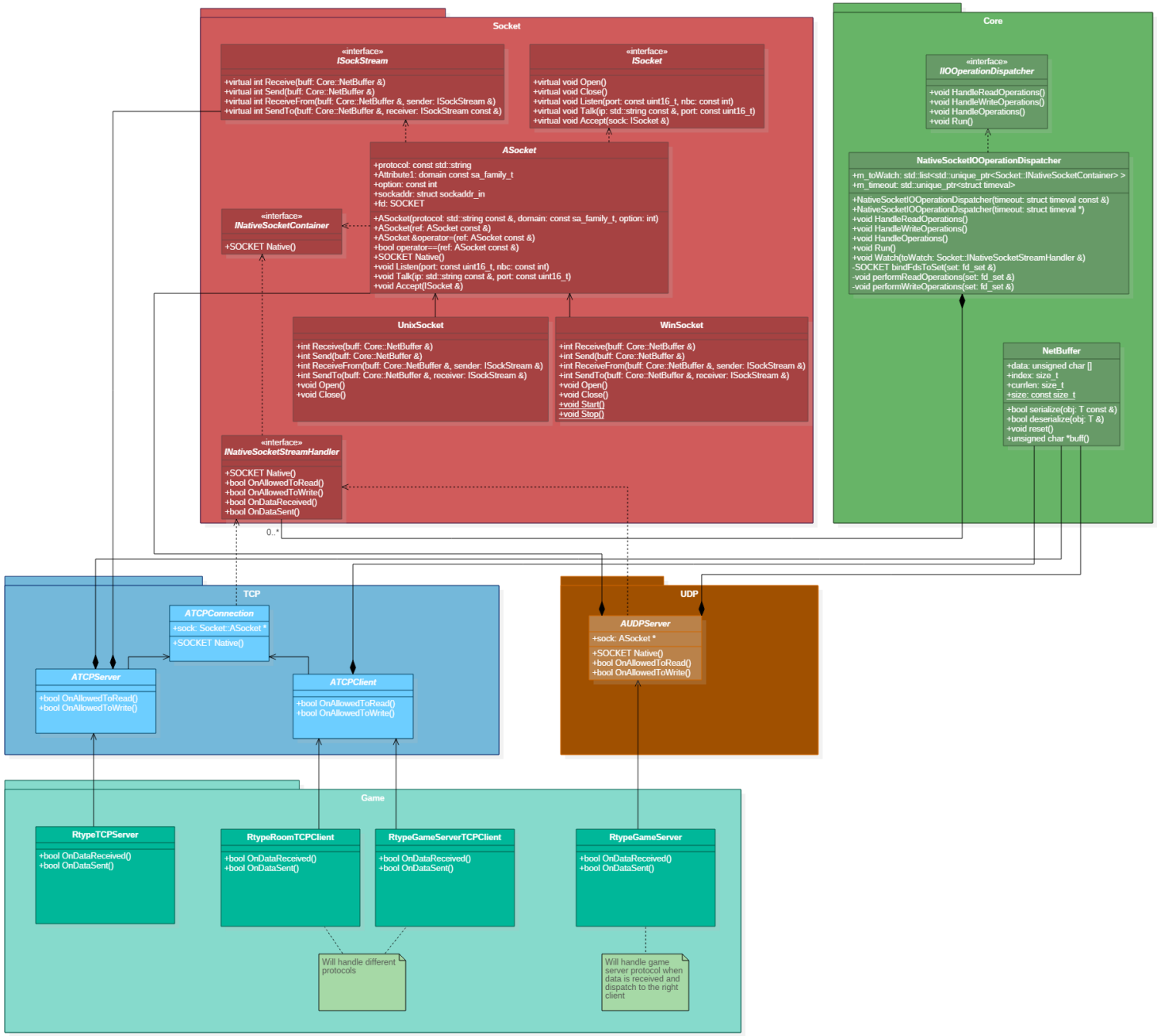

interaction Jouer
PC
ClientGame
ServeurGame
Game
1 : getInput( )
2 : sendAction(action)
4 : checkIfActionIsPossible( )
3 : visualPrediction( )
5 : isActionPossible( )
6 : isActionDone( )

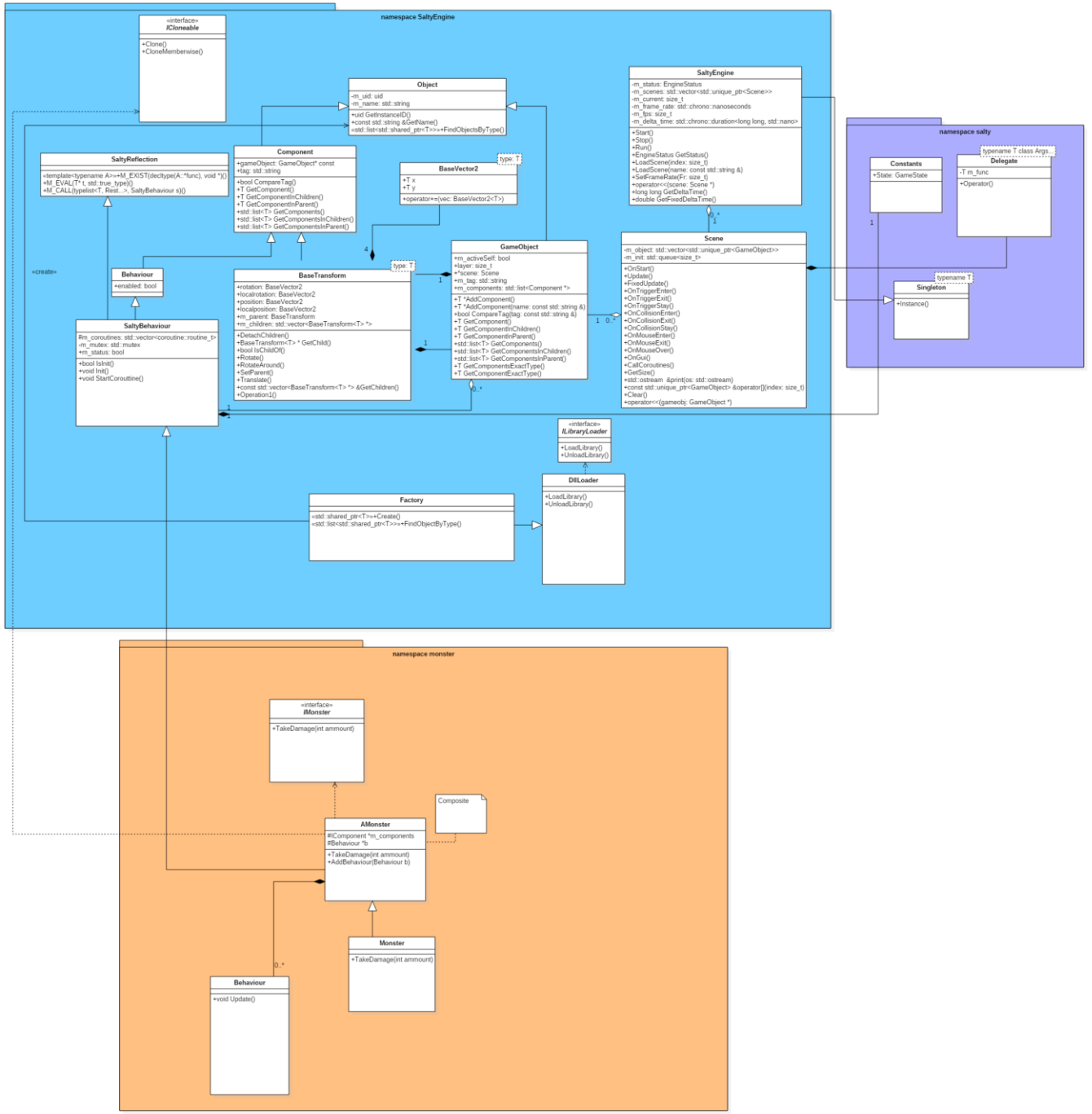

interaction Identification
ClientRoom
ServeurRoom
Identification
1 : sendUsername( )
2 : addUser(username)
3 : checkIfExists(username)
4 : isUserAdded( )
5 : Authenticate( )
6 : roomList( )

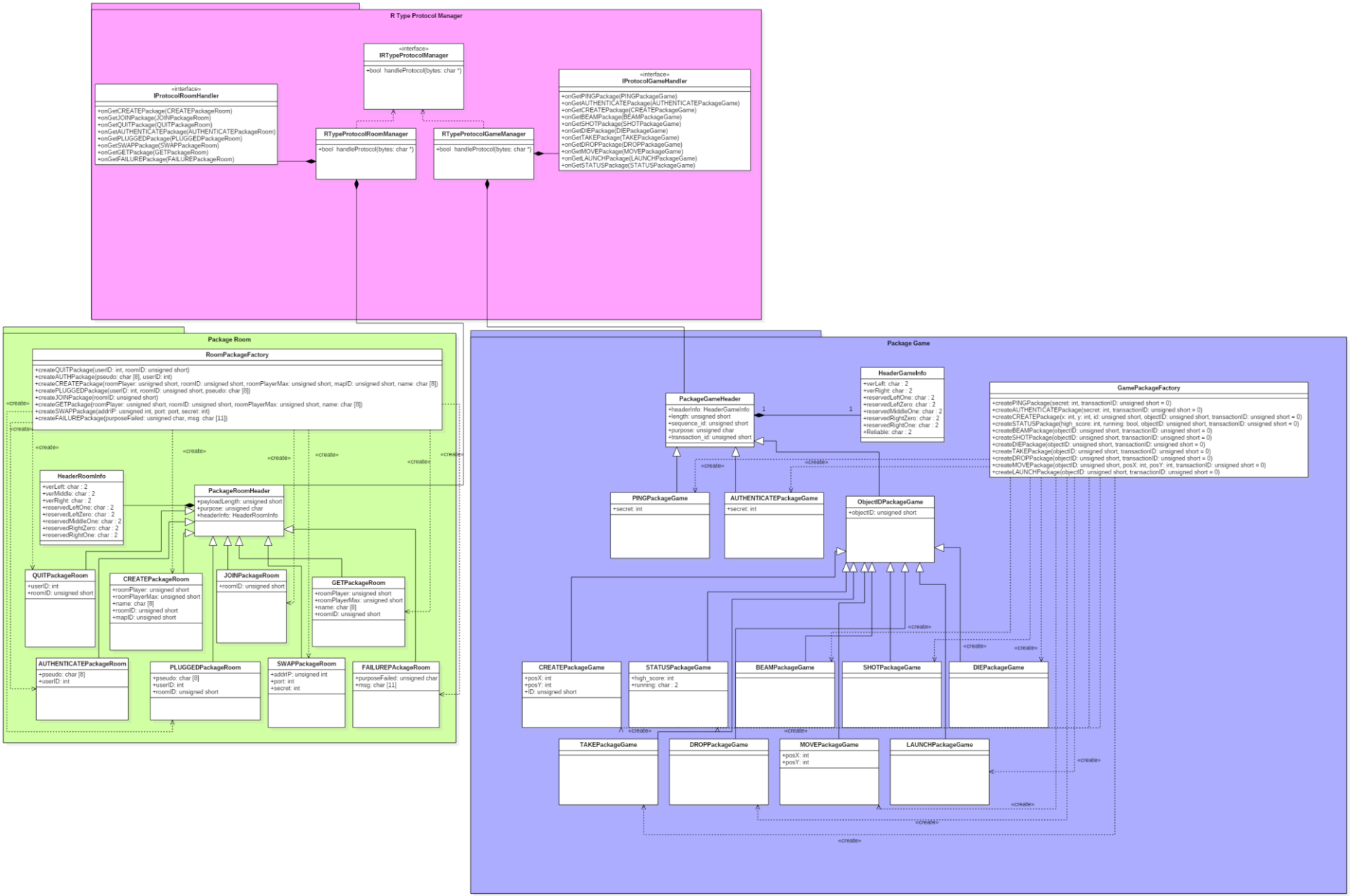

interaction SaltyEngine
SaltyEngine
Scene
GameObject
SaltyBehaviour
SaltyReflection
IRenderer
loop Loop
1 : AddComponent
2 : Component Added
3 : Add GameObject
4 : Object Added
5 : Add Scene
6 : Scene Added
7 : Run
loop Loop
8 : OnStart
alt CheckExist
9 : CheckExist
10 : Start
11 : Start
12 : End Start
13 : End Start
14 : FixedUpdate
15 :
16 : OnTriggerXXX
17 :
18 : OnCollisionXXX
19 :
20 : OnMouseXXX
21 :
22 : Update
23 :
24 : CallCoroutines
25 :
26 : OnGui
27 :
28 : Draw
29 : End Draw

Class Diagrams




Socket
«interface»
ISockStream
+virtual int Receive(buff: Core::NetBuffer &)
+virtual int Send(buff: Core::NetBuffer &)
+virtual int ReceiveFrom(buff: Core::NetBuffer &, sender: ISockStream &)
+virtual int SendTo(buff: Core::NetBuffer &, receiver: ISockStream const &)
«interface»
ISocket
+virtual void Open()
+virtual void Close()
+virtual void Listen(port: const uint16_t, nbc: const int)
+virtual void Talk(ip: std::string const &, port: const uint16_t)
+virtual void Accept(sock: ISocket &)
ASocket
+protocol: const std::string
+Attribute1: domain const sa_family_t
+option: const int
+sockaddr: struct sockaddr_in
+fd: SOCKET
+ASocket(protocol: std::string const &, domain: const sa_family_t, option: int)
+ASocket(ref: ASocket const &)
+ASocket &operator=(ref: ASocket const &)
+bool operator==(ref: ASocket const &)
+SOCKET Native()
+void Listen(port: const uint16_t, nbc: const int)
+void Talk(ip: std::string const &, port: const uint16_t)
+void Accept(ISocket &)
«interface»
INativeSocketContainer
+SOCKET Native()
UnixSocket
+int Receive(buff: Core::NetBuffer &)
+int Send(buff: Core::NetBuffer &)
+int ReceiveFrom(buff: Core::NetBuffer &, sender: ISockStream &)
+int SendTo(buff: Core::NetBuffer &, receiver: ISockStream &)
+void Open()
+void Close()
WinSocket
+int Receive(buff: Core::NetBuffer &)
+int Send(buff: Core::NetBuffer &)
+int ReceiveFrom(buff: Core::NetBuffer &, sender: ISockStream &)
+int SendTo(buff: Core::NetBuffer &, receiver: ISockStream &)
+void Open()
+void Close()
+void Start()
+void Stop()
«interface»
INativeSocketStreamHandler
+SOCKET Native()
+bool OnAllowedToRead()
+bool OnAllowedToWrite()
+bool OnDataReceived()
+bool OnDataSent()
0..*
Core
«interface»
IIOOperationDispatcher
+void HandleReadOperations()
+void HandleWriteOperations()
+void HandleOperations()
+void Run()
NativeSocketIOOperationDispatcher
+m_toWatch: std::list<std::unique_ptr<Socket::INativeSocketContainer> >
+m_timeout: std::unique_ptr<struct timeval>
+NativeSocketIOOperationDispatcher(timeout: struct timeval const &)
+NativeSocketIOOperationDispatcher(timeout: struct timeval *)
+void HandleReadOperations()
+void HandleWriteOperations()
+void HandleOperations()
+void Run()
+void Watch(toWatch: Socket::INativeSocketStreamHandler &)
-SOCKET bindFdsToSet(set: fd_set &)
-void performReadOperations(set: fd_set &)
-void performWriteOperations(set: fd_set &)
NetBuffer
+data: unsigned char []
+index: size_t
+currlen: size_t
+size: const size_t
+bool serialize(obj: T const &)
+bool deserialize(obj: T &)
+void reset()
+unsigned char *buff()
TCP
ATCPConnection
+sock: Socket::ASocket *
+SOCKET Native()
ATCPServer
+bool OnAllowedToRead()
+bool OnAllowedToWrite()
ATCPClient
+bool OnAllowedToRead()
+bool OnAllowedToWrite()
UDP
AUDPServer
+sock: ASocket *
+SOCKET Native()
+bool OnAllowedToRead()
+bool OnAllowedToWrite()
Game
RtypeTCPServer
+bool OnDataReceived()
+bool OnDataSent()
RtypeRoomTCPClient
+bool OnDataReceived()
+bool OnDataSent()
RtypeGameServerTCPClient
+bool OnDataReceived()
+bool OnDataSent()
RtypeGameServer
+bool OnDataReceived()
+bool OnDataSent()
Will handle different protocols
Will handle game server protocol when data is received and dispatch to the right client


namespace SaltyEngine
«interface»
ICloneable
+Clone()
+CloneMemberwise()
Object
-m_uid: uid
-m_name: std::string
+uid GetInstanceID()
+const std::string &GetName()
+std::list<std::shared_ptr<T>>+FindObjectsByType()
SaltyEngine
-m_status: EngineStatus
-m_scenes: std::vector<std::unique_ptr<Scene>>
-m_current: size_t
-m_frame_rate: std::chrono::nanoseconds
-m_fps: size_t
-m_delta_time: std::chrono::duration<long long, std::nano>
+Start()
+Stop()
+Run()
+EngineStatus GetStatus()
+LoadScene(index: size_t)
+LoadScene(name: const std::string &)
+SetFrameRate(Fr: size_t)
+operator<<(scene: Scene *)
+long long GetDeltaTime()
+double GetFixedDeltaTime()
SaltyReflection
-template<typename A>+M_EXIST(decltype(A::*func), void *)()
+M_EVAL(* t, std::true_type)()
+M_CALL(typelist<T, Rest...>, SaltyBehaviour s)()
Component
+gameObject: GameObject* const
+tag: std::string
+bool CompareTag()
+T GetComponent()
+T GetComponentInChildren()
+T GetComponentInParent()
+std::list<T> GetComponents()
+std::list<T> GetComponentsInChildren()
+std::list<T> GetComponentsInParent()
BaseVector2
type: T
+T x
+T y
+operator+=(vec: BaseVector2<T>)
«create»
Behaviour
+enabled: bool
SaltyBehaviour
#m_coroutines: std::vector<coroutine::routine_t>
-m_mutex: std::mutex
+m_status: bool
+bool IsInit()
+void Init()
+void StartCoroutine()
BaseTransform
type: T
+rotation: BaseVector2
+localrotation: BaseVector2
+position: BaseVector2
+localposition: BaseVector2
+m_parent: BaseTransform
+m_children: std::vector<BaseTransform<T> *>
+DetachChildren()
+BaseTransform<T> * GetChild()
+bool IsChildOf()
+Rotate()
+RotateAround()
+SetParent()
+Translate()
+const std::vector<BaseTransform<T> *> &GetChildren()
+Operation1()
GameObject
+m_activeSelf: bool
+layer: size_t
+*scene: Scene
+m_tag: std::string
+m_components: std::list<Component *>
+T *AddComponent()
+T *AddComponent(name: const std::string &)
+bool CompareTag(tag: const std::string &)
+T GetComponent()
+T GetComponentInChildren()
+T GetComponentInParent()
+std::list<T> GetComponents()
+std::list<T> GetComponentsInChildren()
+std::list<T> GetComponentsInParent()
+T GetComponentExactType()
+T GetComponentExactType()
Scene
-m_object: std::vector<std::unique_ptr<GameObject>>
-m_init: std::queue<size_t>
+OnStart()
+Update()
+FixedUpdate()
+OnTriggerEnter()
+OnTriggerExit()
+OnTriggerStay()
+OnCollisionEnter()
+OnCollisionExit()
+OnCollisionStay()
+OnMouseEnter()
+OnMouseExit()
+OnMouseOver()
+OnGui()
+CallCoroutines()
+GetSize()
+std::ostream &print(os: std::ostream)
+const std::unique_ptr<GameObject> &operator[](index: size_t)
+Clear()
+operator<<(gameobj: GameObject *)
«interface»
ILibraryLoader
+LoadLibrary()
+UnloadLibrary()
DllLoader
+LoadLibrary()
+UnloadLibrary()
Factory
«std::shared_ptr<T>»+Create()
«std::list<std::shared_ptr<T>>»+FindObjectByType()
namespace salty
Constants
+State: GameState
Delegate
typename T class Args..
-T m_func
+Operator()
Singleton
typename T
+Instance()
namespace monster
«interface»
IMonster
+TakeDamage(int ammount)
Composite
AMonster
#IComponent *m_components
#Behaviour *b
+TakeDamage(int ammount)
+AddBehaviour(Behaviour b)
Monster
+TakeDamage(int ammount)
Behaviour
+void Update()


R Type Protocol Manager
«interface»
IRTypeProtocolManager
+bool handleProtocol(bytes: char *)
«interface»
IProtocolRoomHandler
+onGetCREATEPackage(CREATEPackageRoom)
+onGetJOINPackage(JOINPackageRoom)
+onGetQUITPackage(QUITPackageRoom)
+onGetAUTHENTICATEPackage(AUTHENTICATEPackageRoom)
+onGetPLUGGEDPackage(PLUGGEDPackageRoom)
+onGetSWAPPackage(SWAPPackageRoom)
+onGetGETPackage(GETPackageRoom)
+onGetFAILUREPackage(FAILUREPackageRoom)
RTypeProtocolRoomManager
+bool handleProtocol(bytes: char *)
RTypeProtocolGameManager
+bool handleProtocol(bytes: char *)
«interface»
IProtocolGameHandler
+onGetPINGPackage(PINGPackageGame)
+onGetAUTHENTICATEPackage(AUTHENTICATEPackageGame)
+onGetCREATEPackage(CREATEPackageGame)
+onGetBEAMPackage(BEAMPackageGame)
+onGetSHOTPackage(SHOTPackageGame)
+onGetDIEPackage(DIEPackageGame)
+onGetTAKEPackage(TAKEPackageGame)
+onGetDROPPackage(DROPPackageGame)
+onGetMOVEPackage(MOVEPackageGame)
+onGetLAUNCHPackage(LAUNCHPackageGame)
+onGetSTATUSPackage(STATUSPackageGame)
Package Room
RoomPackageFactory
+createQUITPackage(userID: int, roomID: unsigned short)
+createAUTHPackage(pseudo: char [8], userID: int)
+createCREATEPackage(roomPlayer: unsigned short, roomID: unsigned short, roomPlayerMax: unsigned short, mapID: unsigned short, name: char [8])
+createPLUGGEDPackage(userID: int, roomID: unsigned short, pseudo: char [8])
+createJOINPackage(roomID: unsigned short)
+createGETPackage(roomPlayer: unsigned short, roomID: unsigned short, roomPlayerMax: unsigned short, name: char [8])
+createSWAPPackage(addrIP: unsigned int, port: port, secret: int)
+createFAILUREPackage(purposeFailed: unsigned char, msg: char [11])
HeaderRoomInfo
+verLeft: char : 2
+verMiddle: char : 2
+verRight: char : 2
+reservedLeftOne: char : 2
+reservedLeftZero: char : 2
+reservedMiddleOne: char : 2
+reservedRightZero: char : 2
+reservedRightOne: char : 2
PackageRoomHeader
+payloadLength: unsigned short
+purpose: unsigned char
+headerInfo: HeaderRoomInfo
QUITPackageRoom
+userID: int
+roomID: unsigned short
CREATEPackageRoom
+roomPlayer: unsigned short
+roomPlayerMax: unsigned short
+name: char [8]
+roomID: unsigned short
+mapID: unsigned short
JOINPackageRoom
+roomID: unsigned short
GETPackageRoom
+roomPlayer: unsigned short
+roomPlayerMax: unsigned short
+name: char [8]
+roomID: unsigned short
AUTHENTICATEPackageRoom
+pseudo: char [8]
+userID: int
PLUGGEDPackageRoom
+pseudo: char [8]
+userID: int
+roomID: unsigned short
SWAPPackageRoom
+addrIP: unsigned int
+port: int
+secret: int
FAILUREPackageRoom
+purposeFailed: unsigned char
+msg: char [11]
Package Game
PackageGameHeader
+headerInfo: HeaderGameInfo
+length: unsigned short
+sequence_id: unsigned short
+purpose: unsigned char
+transaction_id: unsigned short
HeaderGameInfo
+verLeft: char : 2
+verRight: char : 2
+reservedLeftOne: char : 2
+reservedLeftZero: char : 2
+reservedMiddleOne: char : 2
+reservedRightZero: char : 2
+reservedRightOne: char : 2
+Reliable: char : 2
GamePackageFactory
+createPINGPackage(secret: int, transactionID: unsigned short = 0)
+createAUTHENTICATEPackage(secret: int, transactionID: unsigned short = 0)
+createCREATEPackage(x: int, y: int, id: unsigned short, objectID: unsigned short, transactionID: unsigned short = 0)
+createSTATUSPackage(high_score: int, running: bool, objectID: unsigned short, transactionID: unsigned short = 0)
+createBEAMPackage(objectID: unsigned short, transactionID: unsigned short = 0)
+createSHOTPackage(objectID: unsigned short, transactionID: unsigned short = 0)
+createDIEPackage(objectID: unsigned short, transactionID: unsigned short = 0)
+createTAKEPackage(objectID: unsigned short, transactionID: unsigned short = 0)
+createDROPPackage(objectID: unsigned short, transactionID: unsigned short = 0)
+createMOVEPackage(objectID: unsigned short, posX: int, posY: int, transactionID: unsigned short = 0)
+createLAUNCHPackage(objectID: unsigned short, transactionID: unsigned short = 0)
PINGPackageGame
+secret: int
AUTHENTICATEPackageGame
+secret: int
ObjectIDPackageGame
+objectID: unsigned short
CREATEPackageGame
+posX: int
+posY: int
+ID: unsigned short
STATUSPackageGame
+high_score: int
+running: char : 2
BEAMPackageGame
SHOTPackageGame
DIEPackageGame
TAKEPackageGame
DROPPackageGame
MOVEPackageGame
+posX: int
+posY: int
LAUNCHPackageGame
1
1
«create»