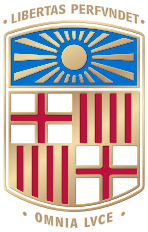
Treball final de grau

GRAU D'ENGINYERIA INFORMÀTICA

Facultat de Matemàtiques i Informàtica



UNIVERSITAT DE BARCELONA

Món 3D per a classes virtuals

Autor: Carla Montserrat Pagès

Director: Ricardo Jorge Rodrigues Sepúlveda Marques
Realitzat a: Departament de Matemàtiques i Informàtica

Barcelona, 20 de juny de 2021

Abstract

The actual situation and the recent health crisis has lead to a reality where daily activities had to adjust to a virtual lifestyle with virtually no time, making some of them more complex than usual. An example of it is attending to class, especially for children, for whom spending hours in front of a laptop, without any kind of human interaction, has become extremely difficult. In this project, the basis for a virtual engine for teachers and students to attend class in a realistic way has been established. To do so, a client-server architecture has been implemented, including a virtual world in which students and teachers can connect and live an experience similar to the real one. The final version includes a set of features which allows the users to perform most of the tipical actions in face-to-face teaching, in a more realistic way than classic video calls and leading to possible extensions that could add more features and also achive a complet realism.

Resum

Degut a la situació actual i la recent crisi sanitària, les activitats quotidianes s'han hagut de convertir a un model virtual sense temps de maniobra. Això ha fet que algunes d'elles, com assistir a classe, s'hagin convertit en una tasca més complicada de l'habitual. Sobretot pels més petits, als qui assentar-se davant d'un ordinador, a escoltar hores de classe sense interacció amb els seus companys, els ha resultat molt complicat. Al llarg d'aquest treball, s'han establert les bases d'una eina virtual que ha de permetre tant a professors com alumnes assistir a classe de la forma més realista possible. Per tal de fer-ho s'ha implementat una arquitectura client-servidor que disposa d'un món virtual on alumnes i professors poden connectar-se i viure una experiència que s'assimila a la que viurien anant a l'escola. En la versió final d'aquest projecte, s'ha obtingut un producte viable que permet realitzar la majoria de les opcions pròpies de l'ensenyament presencial d'una forma més interactiva i realista que les vídeo-trucades tradicionals, donant peu a una ampliació que pugui ampliar les funcionalitats i aconseguir un realisme complet.

Agraïments

Al Ricardo, el meu tutor, pel seu compromís al llarg del procés, per haver-me guiat i aconsellat i per haver aportat sempre idees noves i punts de vista diferents.

A tots els amics i familiars que m'han animat a seguir endavant i m'han donat forces quan ja no me'n quedaven.

Però sobretot, a l'Aida i al Guillem. Sense el seu suport i ajuda incondicionals això no hauria estat possible.

Índex de figures

Índex de taules

Index d'algoritmes

Índex

1. Introducció

1.1 Context i motivació

Les eines de comunicació virtual estan basades en el fet de transmetre imatges i sons a través de cables. La primera referència a una trucada amb vídeo data del 1927 [1] pels laboratoris AT&T Bell Telephone després de 12 anys dels primers experiments i conceptes [2]. Per a fer la demostració van construir una pantalla amb llums de neó que tenia 2500 ànodes, juntament amb una estació de ràdio (Figura 1.1) [3]. De tota manera, no va ser fins al 2003 quan les vídeo-trucades van estar disponibles en la majoria de programes de missatgeria. Des de la seva primera introducció fins a l'actualitat, la comunicació virtual ha anat incrementat la seva presència en tots les camps de treball, com ara tele-cirurgies en l'àmbit mèdic [4].

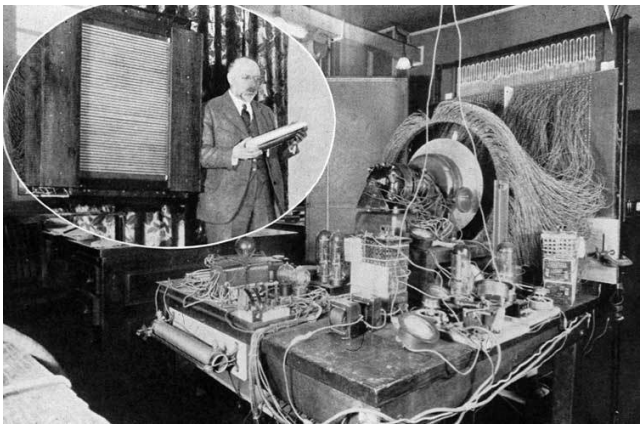


Figura 1.1: Sistema televisiu de la primera vídeo-trucada realitzada per Bell Telephone Laboratories. La imatge principal fa referència a la maquinària, mentre que la inserció de dalt a l'esquerra mostra la pantalla. Imatge extreta de [3].

Recentment, a causa de la pandèmia causada per la COVID-19, el seu ús ha incrementat de manera substancial. Degut a l'obligació de treballar des de casa i la prohibició de reunions presencials, les plataformes per a reunions virtuals han resultat una eina indispensable. A nivell mundial, les descàrregues d'aquestes van arribar a augmentar en un 90% comparat amb la mitja abans de la pandèmia[5].

Aquesta situació no només ha revolucionat l'entorn laboral sinó que també ho ha fet en l'educatiu, on la presencialitat pràcticament ha desaparegut, especialment a les

universitats, i les classes i els exàmens virtuals s'han convertit en la nova modalitat d'educació.

Malauradament, les plataformes disponibles per a reunions virtuals, no estan dissenyades ni pensades per a activitats educatives. Com a conseqüència, no proporcionen l'entorn més favorable per tal de capturar i mantenir l'atenció de l'estudiant. És en aquestes situacions on es veu la necessitat d'una plataforma unificada que pugui proporcionar un millor suport i una millor formació tant a les institucions, com al personal i alumnes, augmenti la motivació i l'atenció i millori l'experiència d'estudiants i professors.

Actualment existeixen algunes plataformes que permeten realitzar totes aquestes accions, ja sigui a través de mons virtuals o simplement a través d'una comunicació on-line i en directe. Tot i així, pel que fa al primer grup, la majoria són tant àmplies i obertes que perden de vista l'objectiu inicial, que és l'educació i l'aprenentatge, fet que pot conduir a distraccions, dificultant així la seva finalitat. D'altre banda, pel que fa al segon grup, hi trobaríem a faltar el component que permet a l'usuari interaccionar amb un món que s'assembli a la realitat.

1.2 Objectius

Tenint en compte els diferents aspectes mencionats anteriorment, l'objectiu principal d'aquest projecte és desenvolupar una plataforma virtual que unifiqui els avantatges de la comunicació on-line amb el realisme dels mons virtuals per tal de que alumnes i professors puguin assistir a classe de la manera més realista i propera possible, augmentant així la motivació d'ambdues parts.

Amb aquesta plataforma es faria més fàcil l'adaptació de l'aprenentatge per als més petits a aquesta nova modalitat d'educació, als qui potser els costa més mantenir l'atenció davant d'un ordinador.

Per tal d'aconseguir-ho es vol implementar una arquitectura client-servidor que proporcioni un entorn on alumnes i professors puguin connectar-s'hi en forma d'avatar i interaccionar amb un món virtual, que simularia l'escola o la facultat. Al final d'aquest període, es pretén haver desenvolupat les bases de l'eina que en un futur podria simular una escola de forma completament realista.

Objectius específics

Els objectius d'aquest projecte es poden dividir en dues parts. En primer lloc es troben els objectius a nivell personal, els quals fan referència al creixement personal, professional i intel·lectual a adquirir durant el projecte. Aquests objectius estan centrats sobretot en l'aprofundiment i el descobriment d'alguns camps de la informàtica, especialment els que m'han generat més interès al llarg de la carrera.

Així doncs, seguint aquesta línia, podem definir els següents objectius:

- Aprendre a planejar i encarar un projecte des de l'inici.
- Explorar el software distribuït.

- Aprendre a realitzar un projecte en Unity.

En segon lloc tenim els objectius centrats en els usuaris de l'aplicació, és a dir, professors i alumnes. Aquests son els inclouen els beneficis que aquest projecte pugui arribar a aportar a les persones que l'utilitzin.

Els objectius a destacar en aquest cas serien:

- Proporcionar una manera més fàcil de fer classes virtuals.
- Maximitzar l'adaptació als nous models d'aprenentatge.
- Maximitzar la motivació i la captació d'atenció de l'alumnat.

1.3 Planificació

Per tal de poder planificar de manera adequada el temps, és necessari definir un seguit de requisits per a crear diferents blocs de treball i tasques que permetin assignar un període de temps per a cada una d'elles.

Per a definir els blocs de treball, primer de tot cal identificar els diferents requisits per dur a terme el projecte. Aquests requisits es poden dividir en diferents grups: requisits essencials, necessaris per a assolir els objectius establerts, i requisits secundaris, aquells que afegiran valor i milloraran el projecte.

Addicionalment, es troben els pre-requisits o requisits bàsics. Aquests són els que fan referència a les funcionalitats principals i necessàries per començar el projecte sobre les quals afegir-hi la resta de requisits.

Requisits

Requisits bàsics

Els requisits bàsics són les funcions primàries necessàries per tal de poder afegir-hi les altres funcionalitats.

- **Avatar i moviment [RB1]**

  Desenvolupar un avatar que simuli a una persona i que sigui capaç de moure's per un món.

- **Edifici i aules [RB2]**

  Crear l'edifici i les aules corresponents on entrar i poder assistir a classe.

- **Multi-Client [R3]**

  Per aquest projecte serà essencial disposar d'un sistema multi-client on tots els usuaris puguin veure's entre ells.

Requisits essencials

Els requisits essencials, per tal de considerar el projecte com a vàlid, són aquells que fan referència a la funcionalitat de fer i assistir a classe. En aquesta línia es plantegen les següents tasques:

- **Assistir a classe [RE1]**

  Correspon a la connexió a una aula en concret. Un cop l'usuari s'hagi connectat a la classe l'alumne podrà interaccionar amb aquesta i podrà accedir a la pissarra i les explicacions del professor obrint la pissarra.

- **Fer classe [RE2]**

  Correspon a la implementació de les funcionalitats necessàries per tal de que el professor pugui accedir a les eines per compartir la veu i les imatges.

- **Compartició de veu (per part del professor) [RE3]**

  Necessàriament el professor haurà de poder compartir veu amb els alumnes.

- **Compartició de diapositives [RE4]**

  El professor podrà compartir contingut en temps real amb els alumnes per tal de dur a terme la classe amb recursos visuals.

  Per aquesta funcionalitat el professor haurà de penjar el contingut en forma d'imatges. Més endavant es valorarà l'opció de que el professor pugui penjar el document en format PDF.

- **Pissarra digital [RE5]**

  Proporcionarà al professor la possibilitat d'escriure en una pantalla en blanc o bé sobre les diapositives que estan essent compartides.

  En aquest cas, les funcionalitats essencials seran escriure en un sol color i amb un sol tipus d'eina amb l'opció d'afegir-ne més, tan d'una com de l'altra.

- **Xat [RE6]**

  Fa referència a la via de comunicació alumne-professor. El xat permetrà que els alumnes puguin escriure dubtes al professor, que seran compartits amb els altres alumnes.

Requisits secundaris

Els requisits secundaris són aquells que no són necessaris per la funcionalitat bàsica del projecte, però que poden millorar-lo.

- **Xat privat [RS1]**

  Aquesta opció permetrà a l'alumne mantenir una conversa privada amb el professor.

- **Tauler [RS2]**

  El tauler serà exclusiu per cada assignatura i es trobarà dins l'aula corresponent. Mitjançant aquest, els alumnes tindran accés als documents utilitzats per al professor.

- **Calendari [RS3]**

  El calendari també serà exclusiu per a cada assignatura i es trobarà dins l'aula corresponent. Els alumnes tindran accés a ell i podran consultar les tasques programades.

- **Compartició de veu (per part dels alumnes) [RS4]**

  Com a valor afegit, els alumnes podran compartir la veu per a preguntar dubtes i intervenir a la classe.

Tasques

Tenint en compte els objectius plantejats inicialment i els requisits del sistema, aquests es distribueixen en diferents blocs de treball per tal d'assignar l'ordre i un període de temps de treball per a cada un.

- **T1: Organització del projecte**

  Planificar com es durà a terme el projecte, quines metodologies es seguiran i quins seran els objectius i requisits.

- **T2: Disseny de l'arquitectura**

  Pensar i dissenyar com seran el client, el servidor i la seva interacció. Quin tipus de comunicació utilitzaran, com seran els paquets que intercanviaran, com es gestionaran cada un ...

- **T3: Desenvolupament de l'arquitectura**

  Implementar el codi bàsic per tal de tenir una comunicació entre un client i un servidor que permeti afegir-hi totes les funcionalitats especificades.

- **T4: Desenvolupament de requisits essencials**

  Desenvolupar els requisits definits com a essencials en l'apartat anterior.

- **T5: Desenvolupament de requisits secundaris i/o millora dels existents**

  Ampliar el sistema ja sigui per afegir-hi funcionalitats o bé per modificar i complementar les ja existents.

- **T6: Anàlisi, resultats i imprevistos**

  Període dedicat a analitzar el funcionament del sistema i a la redacció de la memòria corresponent. També a possibles imprevistos i endarreriments en el desenvolupament dels apartats anteriors.

Taula de planificació (Gantt chart)

A la taula que es mostra a la Figura 1.2 es pot veure quina ha estat la distribució de les tasques en el temps en forma de diagrama de Gantt.

1.4 Organització del document

En aquest document s'explica el procés de creació d'aquest treball final de grau, des del plantejament fins als resultats.

En primer lloc, es contextualitzarà el problema i es parlarà dels motius i la motivació per realitzar-lo. Es definiran uns objectius i també uns requisits i unes tasques que recullin els objectius i ens permetin planificar el temps.

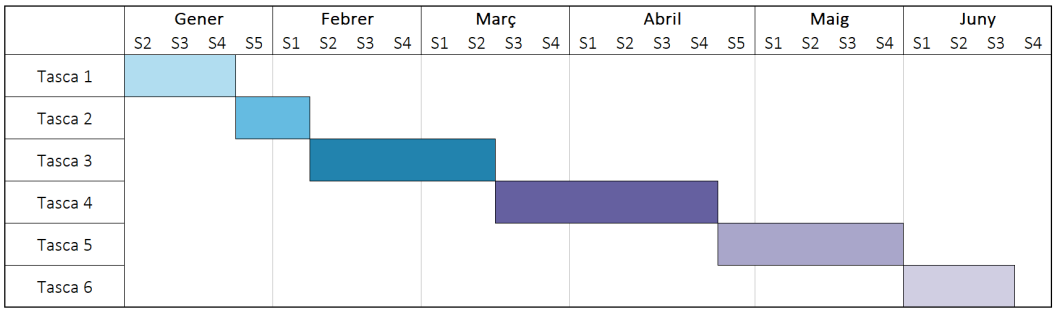


Figura 1.2: Diagrama de Gantt

A continuació, s'explicarà l'anàlisi detallat del problema i les seves parts més tècniques, es plantejarà des del punt de vista de l'usuari i s'exposaran les tecnologies a utilitzar.

S'explicarà també el procés de desenvolupament, és a dir, les decisions de disseny més importants així com la justificació de les tecnologies utilitzades.

S'inclourà també un apartat on s'explicaran quins han estat els resultats del projecte i quins han estat els resultats de les proves d'aquest amb usuaris.

Finalment, es valorarà el treball fet en funció als objectius plantejats per tal de decidir si el projecte ha estat realitzat amb èxit seguit d'un apartat explicatiu de com utilitzar el programa.

2. Estat de l'art

Actualment existeixen diferents plataformes que permeten realitzar accions similars a la que es vol desenvolupar en aquest treball. Es poden distingir dues tipologies de plataformes que compleixen aquests requisits. Per una banda, es troben aquelles que proporcionen mons virtuals que permeten interactuar amb usuaris i realitzar diferents activitats i, per altre banda, les que es limiten única i exclusivament a la comunicació on-line amb opcions de compartir àudio, vídeo i documents.

Mons virtuals

La gamificació s'ha convertit en una estratègia utilitzada per a professors amb l'objectiu d'augmentar la motivació dels estudiants. La gamificació es pot desenvolupar en entorns reals i virtuals [6]. En aquest apartat es descriuen algunes de les plataformes més comunes per al disseny i creació de mons virtuals orientats a l'educació.

The Education District

The Education District és un mon virtual per a facilitar experiències d'aprenentatge digital. Està centrat, principalment, en utilitzar la col·laboració i la gamificació per al desenvolupament de competències i habilitats.[7] Aquesta plataforma permet, entre d'altres, crear mons 3D multi-usuari amb interacció i possibilitat de fer qualsevol tipus de creacions, tot en temps real[8].

Figura 2.1: Visita virtual a un museu a través de The Education District. Imatge reproduïda de [7].

Una de les aplicacions principals són les visites virtuals a llocs reals, com per exemple museus. Aquest tipus d'aplicacions recolza l'educació cultural dels estudiants a la ve-

gada que redueix la planificació i el cost d'aquestes activitats. A més a més, permet fer visites a llocs que presencialment no seria possible, com ara a altres països [9].

OpenWonderland

Open Wonderland és un conjunt d'eines de codi obert en Java que permet crear mons virtuals. Un dels objectius principals al crear la plataforma era la col·laboració i interacció entre usuaris i/o desenvolupadors [10].

Relacionat amb aquest aspecte, es va veure que cada cas podia beneficiar-se d'altres aplicacions o implementacions, així doncs van la plataforma ofereix la possibilitat d'ampliar i personalitzar l'ambient. L'eina està dissenyada per a desenvolupadors que estiguin familiaritzats amb el llenguatge Java ja que permet ampliar qualsevol part del sistema, des del client i el servidor fins a la possibilitat d'afegir-hi noves funcionalitats i nous mons creant mòduls [11, 10]. Un altre dels aspectes interessants d'aquesta eina son la possibilitat de connectar-se a través del telèfon o bé les converses privades [12].

La flexibilitat de la plataforma resulta en aplicacions molt diverses, com ara la creació de tallers i *workshops* ja que en aquests casos la presencialitat limita la quantitat de participants i la transmissió del coneixement. Un exemple concret és la creació de cursos per a l'aprenentatge d'*agile software* [13].

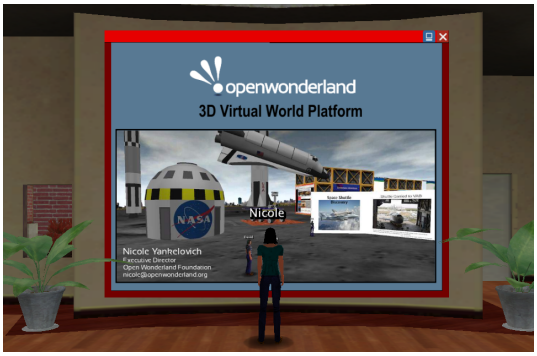


Figura 2.2: Il·lustració d'un mon virtual desenvolupat amb Open Wonderland. Imatge extreta de [14].

SecondLife

Second Life és un mon virtual 3D amb més de 10 anys d'història al que s'hi pot accedir lliurement a través del propi programari de Linden Lab o mitjançant visors alternatius de tercers [15].

Els usuaris creen representacions virtuals d'ells mateixos, anomenats avatars, i poden interactuar amb espais, objectes i altres avatars. Poden explorar el món (conegut com a quadrícula), conèixer altres residents i participar en activitats lúdiques, socials o polítiques.

L'univers té el seu propi sistema econòmic: els residents poden crear i vendre objectes, propietats o serveis. Els intercanvis es fan en dòlars Linden, una moneda virtual que pot ser intercanviada per monedes reals. [16]

Second Life és una de les plataformes líders en el mon de l'educació [17]. Un exemple a destacar és la implementació d'un laboratori de química per a pràctiques de laboratori i experiments. Facilita l'aprenentatge dels estudiants en temes complexos, com ara els models atòmics, ja que permet una visualització 3D de la matèria d'estudi [18].

Figura 2.3: Comparació d'un laboratori de química real amb un ambient creat amb SecondLife. Imatge reproduïda de [17].

OpenSimulator

Open Simulator és un servidor de codi obert per a crear ambients i mons virtuals en 3D. Està desenvolupada en C# i és un esforç comú dels membres de la comunitat [19]. Permet customitzar els mons virtuals utilitzant qualsevol tipus de tecnologia que s'adapti a l'usuari [20]. A més a més, ofereix la possibilitat d'expandir l'ambient de manera senzilla a base de mòduls [19] i és compatible amb el client de Second Life [21].

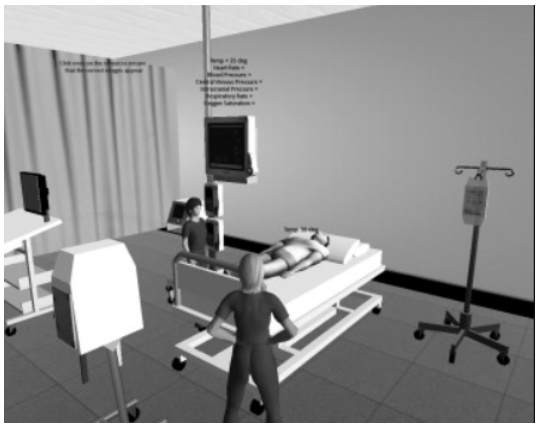


Figura 2.4: Il·lustració de dos infermers/es examinant un pacient en un ambient desenvolupat mitjançant Open Simulator. Imatge extreta de [22].

Tot i que és una plataforma encara en desenvolupament, és força estable [20]. Una característica interessant d'aquesta plataforma és el fet de poder connectar múltiples regions independents, que permet connectar diferents mons virtuals [23].

Una aplicació d'Open Simulator és l'ensenyament de cures intensives, ja que en aquests casos l'educació té un cost molt elevat, en temps i econòmicament Figura 2.4. La implementació de plataformes virtuals per a assistir el procés d'ensenyament, especialment en situacions d'alt risc, reforça i millora la qualitat de la formació d'infermers/es [22].

Plataformes de vídeo-trucades

En el camp de les comunicacions virtuals es poden trobar un gran ventall d'aplicacions que permeten reunir a grups de persones i compartir tot tipus de continguts. Els principals aspectes que diferencien les opcions disponibles són la quantitat de persones que poden acomodar, el cost de la plataforma o la possibilitat d'interacció per pantalla de l'usuari, com ara aixecar la mà o aplaudir. Tot i que la finalitat de la majoria de plataformes és la mateixa, no totes estan orientades o utilitzades al mateix entorn [24], tal i com es pot observar en la llista i figura (Figurea **??**) següents:

- En l'àmbit **empresarial** les plataformes més utilitzades són Skype for Business, Slack, Cisco WebEx i Micorsoft Teams.
- Pel que fa a l'**educació** algunes les més comunes són Zoom, Google Meet, BlackBoard Collaborate i Jitsi meet.
- Finalment, en un entorn **social i d'oci** les més habituals són Jitsi meet, Whereby, Discord i Twitch.

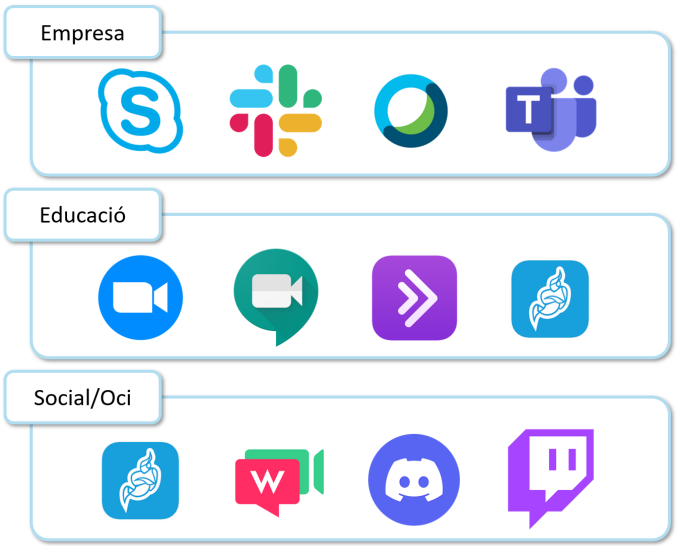


Figura 2.5: Exemples d'algunes plataformes per a vídeo-trucades classificades segons l'àmbit en el que són més utilitzades.

3. Anàlisi del problema

En aquest capítol es detalla el plantejament del projecte. S'explica des de quines son les necessitats dels usuaris fins a quines son les necessitats del sistema. Es detallà com serà l'arquitectura del programa i quins problemes ens podem trobar al llarg del procés de desenvolupament.

3.1 Usuaris i casos d'ús

En aquest projecte es defineixen dos tipus d'usuaris, els alumnes i els professors, que seran els principals consumidors de la plataforma.

Els objectius del projecte han estat dividits en tres categories segons la importància del cas d'ús: bàsics, essencials, i secundaris. Després de plantejar-los i conceptualitzar breument les implicacions de cada un, es va optar per inicialment desenvolupar només aquells que constituïen l'eix cèntric de la plataforma. D'aquesta manera, es pot garantir arribar a un producte viable mínim i, incloent els essencials, a una eina plenament funcional. Un cop s'hagin desenvolupat correctament aquests casos, es valorarà la opció d'implementar els que corresponen als requisits secundaris en funció del temps restant.

Casos d'us compartits

Com que la majoria dels casos d'us són compartits, en aquesta secció s'expliquen els que s'apliquen a ambdós tipus d'usuaris.

- **[UC1] Log In:** El professor ha de poder iniciar sessió a la plataforma.
- **[UC2] Escollir avatar:** El professor ha de poder escollir quin és l'avatar que el representa.
- **[UC3] Entrar a l'escola**: El professor ha de poder entrar a l'escola.
- **[UC4] Observar el món:** El professor ha de poder veure com els usuaris connectats interaccionen amb el món.
- **[UC5] Interaccionar amb el món:** El professor ha de poder-se moure lliurement dins el món i interaccionar-hi.
- **[UC6] Entrar a l'aula**: El professor ha de poder entrar a l'aula.

Professor

- **[UC7] Escriure en una pissarra:** El professor ha de poder disposar d'un espai on poder escriure les explicacions que vulgui fer a l'alumnat.
- **[UC8] Compartir diapositives:** El professor ha de poder compartir recursos amb els seus alumnes.
- **[UC9] Compartir veu:** El professor ha de tenir la opció de compartir la seva veu per tal de que els alumnes puguin escoltar les seves explicacions.
- **[UC10] Escriure en un xat:** El professor ha de poder interactuar de forma escrita amb els alumnes.

A la Figura 3.1 podem veure el diagrama de casos d'ús pel cas dels professors.

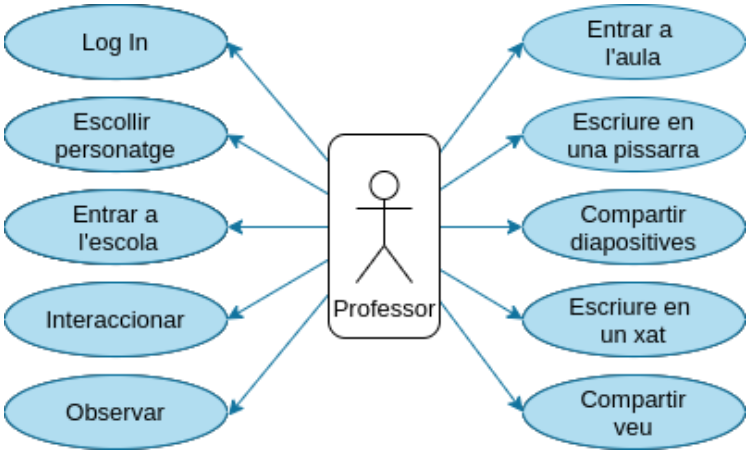


Figura 3.1: Diagrama de casos d'ús pels professors

Alumne

- **[UC7] Visualitzar una pissarra:** L'alumne ha de poder disposar d'un espai on veure el tot allò que està escrivint i compartint el professor.
- **[UC8] Escoltar explicacions:** L'alumne ha de poder escoltar les explicacions que fa el professor.
- **[UC9] Escriure en un xat:** L'alumne ha de poder interactuar de forma escrita amb el professor per tal de fer consultes.

A la Figura 3.2 podem veure el diagrama de casos d'ús pel cas dels professors.

3.2 Arquitectura del sistema

3.2.1 Visió general del Sistema

Per realitzar aquest projecte, s'ha optat per utilitzar un sistema client-servidor. Aquesta arquitectura és preferible a una arquitectura peer-to-peer (o P2P) ja que per aquesta

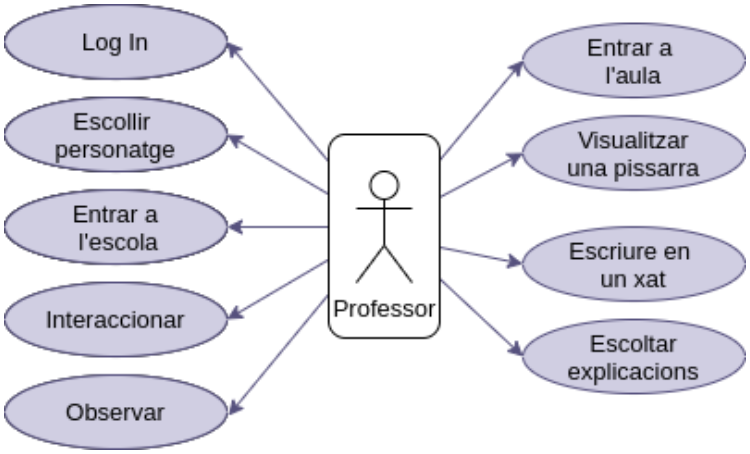


Figura 3.2: Diagrama de casos d'ús pels alumnes.

segona es necessita buscar la IP a la que ens volem connectar, fent que el sistema sigui més complex. En el cas bàsic on tots els ordinadors estiguin a la mateixa xarxa, aquest problema es podria atacar de forma relativament senzilla, però la dificultat incrementa exponencialment segons els dispositius es trobin connectats a altres xarxes. Per exemple, es podria donar el en cas que el professor estigui connectat per ethernet i els alumnes en canvi hi estiguin per wifi, on s'hauria de passar pel canvi d'IP que aquesta genera, o bé podria existir un firewall que, d'entrada, impedís fer aquesta connexió i compliqués molt trobar-ho.

Per contra, un servidor facilita utilitzar una IP estàtica, de forma que no faci falta cap procés de descobriment. Addicionalment, els servidors, generalment, estan sempre engegats. Això permetria accedir a l'escola sempre que es volgués, fomentant trobades fora de l'horari per avançar feina conjuntament. A banda d'això, l'arquitectura client-servidor centralitza totes les dades en el servidor. Això ens garantitza un seguit d'avantatges, com per exemple:

- Les dades sempre estan disponibles, independentment de l'ordinador que les hagi creat. Això, que aparentment podria no tenir moltes implicacions, significa que diferents professors podrien modificar una mateixa aula des d'ordinadors i moments diferents
- Possibilita l'ús de rèpliques, en cas que es vulgui tenir la informació duplicada o triplicada per evitar-ne la pèrdua
- En cas de necessitar una migració, per exemple, un canvi de versió que introdueix millores, els procés d'actualitzar les dades es pot fer transparent i indolor pels clients

Per contra, en l'arquitectura P2P les dades estan distribuïdes entre els diferents terminals i dificulta o inclús impossibilita algunes de les opcions abans anomenades.

Per tal de realitzar aquest projecte es necessiten tres components essencials: (1) un client que capturarà les interaccions que vulgui fer l'usuari, (2) un servidor que gestio-

narà aquestes interaccions i (3) una base de dades que s'encarregarà d'emmagatzemar la informació que hagi de ser permanent. Tal i com es veu a la Figura 3.3.

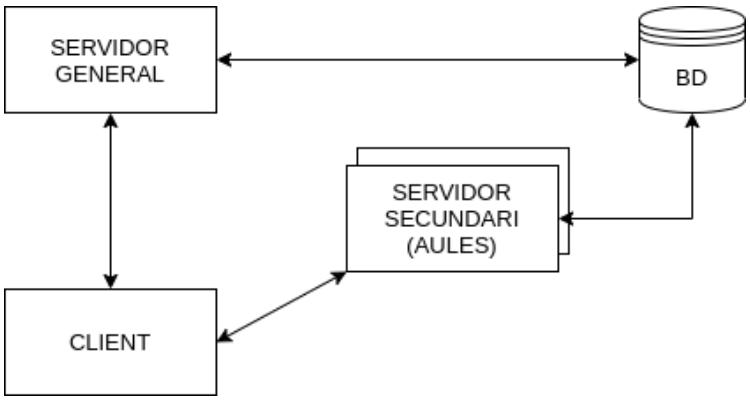


Figura 3.3: Diagrama de l'arquitectura del sistema.

A més a més, a part de necessitar un protocol d'intercanvi de missatges, també serà necessari establir un protocol de connexió entre el client i el servidor. En aquest cas, s'utilitzarà el protocol TCP (Transmission Control Protocol o Protocol de control de transmissió). Aquest protocol ens garanteix que tots els paquets que s'enviïn seran entregats de manera ordenada i sense perdre informació, a diferència d'altres protocols com UDP (User Datagram Protocol o Protocol de datagrames d'usuari) que garanteixen rapidesa i una comunicació sense necessitat de connexió, però no asseguren l'ordre ni una recepció completa.

Així doncs,tenint en compte que per la majoria d'accions d'aquest projecte és necessari rebre tota la informació que s'ha enviat, i a més a més en l'ordre correcte i davant de totes les avantatges que hem comentat anteriorment que aquest protocol ens ofereix, s'ha optat per utilitzar el protocol TCP.

És important també mencionar l'aspecte de la seguretat, doncs és una peça fonamental de tot sistema informàtic. Per una banda, les eines utilitzades (Unity) garanteixen que el programa serà segur i no provocarà problemes al sistema de l'usuari. D'altra banda, el protocol que dissenyarem és suficient per garantir que els paquets que s'enviaran i rebran no podran fer res més que el que estan explícitament dissenyats per fer.

En jocs on-line, és comú l'explotació del protocol per utilitzar comportaments no programats. Per exemple, en jocs d'acció, es podria modificar il·legalment el protocol, o inclús el joc, de forma que el teu personatge fos invencible.

Tenint en compte el públic al que està destinada la plataforma, és a dir, alumnes de primaria i ESO, la seguretat davant possibles trampes no resulta en una prioritat. D'entrada, es poc probable que tinguin el coneixement necessari per fer cap d'aquestes modificacions fora del comportament normal. I, inclús si les tinguessin, es tracta d'un entorn limitat on no podrien introduir cap canvi significatiu. Alguns dels canvis que podrien arribar a fer serien en el seu nom d'usuari o bé, en casos molt remots, arribar a aconseguir accés per escriure a la pissarra.

Davant de la inversió que suposaria en temps fer un protocol criptogràficament segur, incloent la privacitat de la comunicació, s'ha decidit decantar la balança cap a la implementació de més característiques útils per l'ensenyança. Característiques a les que se'ls pugui aplicar millores, obtenint un sistema el màxim complert possible.

3.2.2 Servidor

El servidor és l'encarregat de gestionar els clients. S'encarrega d'acceptar les seves connexions, gestionar les peticions que aquests faran i, en aquest cas, també de mantenir-los informats de quina és la situació dels altres clients connectats en aquell moment. A més a més, un servidor també pot realitzar connexions a bases de dades i les consultes corresponents o bé gestionar altres tipus de controls sobre el sistema, com per exemple col·lisions amb NPC.

En aquest projecte, quan es parli de servidor podrem fer referència tant a l'escola com a les aules, ja que cada un d'aquests estarà gestionat per un servidor diferent. El servidor amb el que els alumnes tindran un primer contacte és el que representarà l'escola. A part d'això, com s'ha comentat anteriorment, cada aula estarà gestionada per separat. Aquestes podran estar en un servidor completament diferent o en el mateix però en un procés diferent i els usuaris s'hi podrà connectar a través del servidor principal, l'escola.

El servidor escola haurà de tenir constància en tot moment de quina quantitat de servidors secundaris hi ha, mentre que les aules només necessitaran saber que hi ha un servidor principal al que poder tornar quan se surti de l'aula.

Tots els servidors tindran un llistat de clients independent l'una de l'altre. Això garanteix un sistema més eficient ja que cada servidor només gestionarà la llista de clients corresponents al seu espai, en comptes de tenir un sol servidor que, a l'hora de gestionar peticions i interaccions, hagi de considerar els espais on estan els clients i a qui van destinades les peticions. És per aquest motiu que totes les aules s'estaran executant en processos separats del general i d'entre elles mateixes.

Tots els servidors, independentment de si es tracta de l'escola o d'una aula, es plantejaran en dues classes essencials: per una banda una classe que s'encarregarà d'acceptar les connexions dels servidors i de crear un fil per a cada client per tal de gestionar-los individualment i per altre banda, tindrem la classe que executaran aquests fils, que inclourà tota la lògica de la connexió.

A més a més, els servidors també s'encarregaran d'establir la connexió amb la base de dades ja que serà el que gestionarà totes les accions dels clients i, per tant, serà l'únic que necessitarà accés a les dades.

3.2.3 Client

Un client és un programari que accedeix a un servidor per tal de fer-hi peticions i n'espera la seva resposta. Aquestes peticions seran les que sorgeixin de recollir les interaccions dels usuaris.

El client haurà de disposar d'alguns components visuals per tal d'aconseguir el realisme que busquem en el projecte. Aquests components son un avatar que pugui represen-

tar als usuaris, un entorn que simuli l'escola o les aules, i d'altres espais de similar naturalesa que ens puguin ajudar a aconseguir el realisme desitjat.

Per tal de proporcionar als usuaris totes les funcionalitats plantejades a l'inici, s'hauran de buscar maneres gràfiques el màxim intuïtives possible. Per exemple, fer click a una pissarra per obrir-la, veure'n els continguts i escoltar les explicacions.

Pel que fa referència a la compartició de diapositives, s'ha optat per un model en el que s'han de penjar individualment en format d'imatge de manera que el professor, mitjançant un menú de selecció d'arxius, haurà d'accedir al seu sistema i seleccionar les imatges. Mitjançant aquest procés, les imatges es carregaran a la pissarra al moment, enviat-se també a tots els alumnes connectats. En versions futures de la plataforma es plantejarà l'opció de permetre pujar un PDF directament, de forma que aquest sigui convertit automàticament al format que correspongui.

Com comentàvem anteriorment, el professor necessita una pissarra per poder escriure-hi algunes explicacions. A la implementació, la pantalla serà enviada cada un cert temps a la resta d'alumnes. Per una banda, establir un període curt ens garanteix que els alumnes rebran els canvis amb una gran continuïtat i no apreciaran talls, però suposarà un intercanvi major de paquets i una probabilitat més elevada de saturar el sistema degut a la gran quantitat de dades que aquest estarà intercanviant. Per altre banda, establir un període llarg d'enviament ens pot estalviar aquesta saturació del sistema, però faria que la freqüència amb la que els usuaris reben els canvis sigui molt baixa i, per tant, que els alumnes no puguin seguir correctament les explicacions ja que veurien canvis molt significatius amb un retard considerable. Al llarg del desenvolupament d'aquesta característica s'haurà d'avaluar quin és el millor període d'enviament.

En referència a la compartició de veu, és important també el tema de la continuïtat en la reproducció per tal de garantir que l'alumne entén correctament les explicacions. En aquest cas, ens trobem amb la mateixa problemàtica que ens trobàvem quan parlàvem de la pissarra. Per tal de que l'usuari es mantingui el màxim actualitzat, serà necessari enviar la veu que esta sent enregistrada en petits talls d'àudio al cap d'una quantitat de temps. Aquest temps ha de permetre maximitzar la continuïtat de l'explicació, el mínim delay i la mínima saturació del sistema. Per tal d'aconseguir les mínimes pauses possibles, s'estudiarà la possibilitat de crear un buffer que acumuli una quantitat concreta de talls de veu abans de començar a reproduir-los.

Donat que el protocol que ja tenim està fet sobre TCP i està pensat de forma que sigui extensible i flexible, és molt més senzill implementar una primera versió de la comunicació de veu sobre aquest mateix vehicle que no pas e l'laborar-ne una de nova. Això ens permet un ràpid desplegament d'una nova funcionalitat sense perjudicar el temps invertit en d'altres. Com s'ha comentat anteriorment, aquest producte és una primera versió que esperarà millores en un futur, i per tant permetrà desenvolupar aquesta part del sistema seguin el protocol UDP en un futur.

3.2.4 Base de dades

Una base de dades és un conjunt de dades estructurades i accessible des de diversos programes després de crear una connexió.

Es necessita una base de dades per tal de poder guardar tota aquella informació que hagi de ser permanent. En aquest cas, l'únic que es guardarà a la base de dades seran els usuaris amb les seves respectives contrasenyes, que hauran d'estar codificades mitjançant una funció de hash per tal d'evitar guardar dades sensibles de forma plana, i el rol d'aquests dins el joc.

En aquest apartat disposem de varies tecnologies possibles per a utilitzar. Per una banda, podem escollir entre una base de dades relacional o SQL i una base de dades no relacional o NoSQL. També disposem de diferents sistemes gestors de bases de dades. Per a bases de dades relacionals podríem utilitzar PostgreSQL, MySql o MariaDB entre d'altres. Per a NoSQL en canvi disposaríem d'algunes opcions com ara MongoDB, Redis o Casandra entre d'altres. Més endavant s'avaluaran aquestes opcions per tal d'escollir la que s'adapti a aquest problema.

3.2.5 Protocol

Per tal de realitzar la comunicació amb èxit, també serà necessari desenvolupar un protocol que estableixi com seran els missatges que s'enviaran el client i el servidor i quins seran.

És important també que el protocol sigui ampliable, és a dir, que resulti fàcil afegir-hi nous paquets quan sigui necessari per tal de no haver-lo de reformular sencer.

4. Disseny i desenvolupament

En aquest capítol es parlarà de les decisions de disseny que s'han pres al llarg del procés de desenvolupament del projecte. S'explicarà individualment per cada component del sistema i s'especificaran des de les tecnologies utilitzades fins a com s'ha desenvolupat cada part i com s'han solucionat els problemes que s'havien previst.

4.1 Servidor

Tecnologies utilitzades

En el servidor s'ha utilitzat Java com a llenguatge de programació. Java és un llenguatge amb el que la majoria d'estudiants ens sentim còmodes a l'hora de programar ja que l'hem utilitzat molt al llarg de la carrera. A més, quest llenguatge ofereix molt bon suport per al desenvolupament del sistemes client-servidor i s'ha utilitzat a l'assignatura de Software Distribuït, que és l'assignatura on s'ensenyen nocions sobre aquestes arquitectures, i per tant utilitzar-lo oferia la possibilitat de poder aplicar coneixements adquirits al llarg de la carrera.

Pel que fa a l'editor, s'ha utilitzat VisualStudio Code ja que ofereix molt bon suport a l'hora de desenvolupar en Java però sobretot perquè era la millor opció per al desenvolupament del client i així s'unificaven al màxim les tecnologies utilitzades.

Desenvolupament

Com s'ha comentat en varies ocasions, disposem de dos tipus de servidors: l'escola i les aules. No obstant, independentment de que les funcions que facin siguin diferents, el codi que els executa és el mateix a excepció d'un detall.

Com es pot veure a la Figures 4.1 i a la 4.2, la diferència bàsic entre els dos és la quantitat d'aules que s'instancien. Mentre que el servidor general necessita informació de totes les aules per tal de poder gestionar els canvis, tal com s'explica més endavant, l'aula només necessita informació general sobre ella mateixa.

L'únic efecte que te això en la classe ClientHandler, l'encarregada de gestionar la lògica, és la manera en com aquesta s'instancia. Per afrontar aquest problema disposem de dos constructors, un en el que rebrà un llistat d'aules i un en el que només rebrà una aula. Pel que fa a la resta del codi, és el mateix pels dos casos.

Globalment, el servidor està desenvolupat de forma asíncrona ja que pot gestionar múltiples clients alhora. El que permet que això es pugui fer és la creació de threads per

cada client, que es fa mitjançant la classe ExecutorService de Java. No obstant, la gestió individual de cada client és síncrona ja que fa crides bloquejants a l'enviar i rebre els paquets.

Podem considerar però que el servidor és asíncron ja que, al parlar d'aquest aspecte, ens referim a la sincronia/asincronia a nivell de la possibilitat de gestionar múltiples usuaris, que és exactament el que ens permet fer aquest servidor.

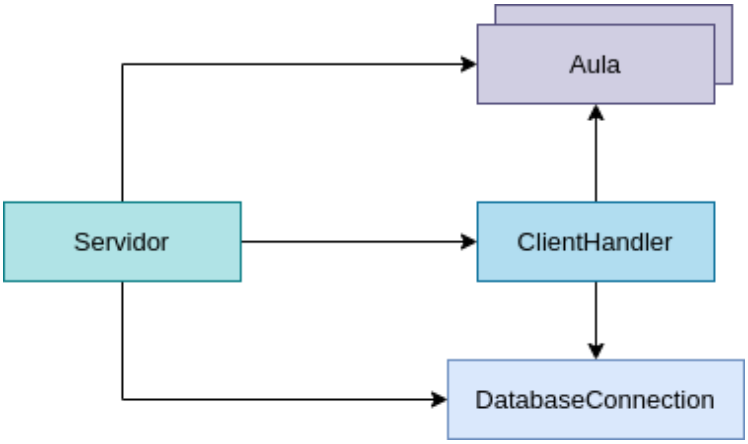


Figura 4.1: Diagrama representatiu de les classes del servidor-escola. En aquest diagrama es mostren les diferents classes que componen el servidor, com s'instancien i com es criden entre elles.

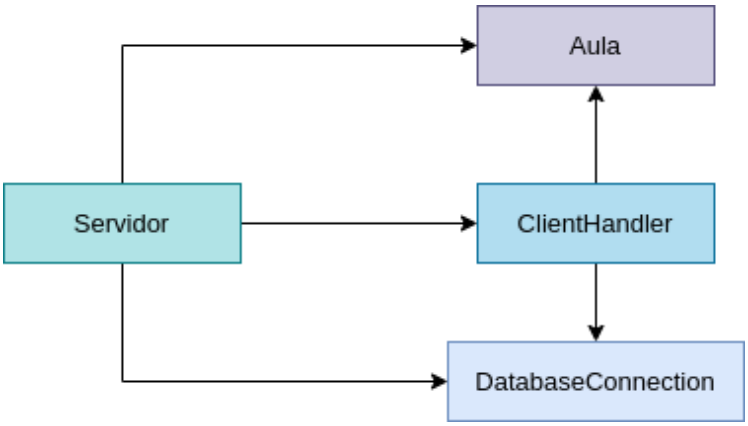


Figura 4.2: Diagrama representatiu de les classes del servidor-aula. Podem veure quines són les classes que componen el servidor i com s'instancien i es criden entre elles.

El servidor també s'encarrega de gestionar el canvi entre els dos espais. Les aules estan creades de manera que guarden la informació més important sobre elles. Abans d'entrar a l'aula, el que ens interessa és la IP i el port del servidor que les executa i també el punt del mapa en el que estan ubicades. Aquest punt es defineix per la posició de la porta. Un cop el servidor rep el punt on el client es vol moure, comprovarà si

aquest punt queda dins de l'àrea que envolta la porta, calculada mitjançant la distància Euclidea, i si aquest és el cas envia al client la nova IP i el nou port per tal de que aquest realitzi el canvi pertinent. Aquest procés s'explica més endavant.

A banda d'aquesta informació, l'aula també guarda informació rellevant pel desenvolupament de l'interior de les classes. En concret, necessita tots els atributs referents a la pissarra. Cada vegada que el servidor rep alguna modificació de la pissarra, a part de reenviar-la a tots els usuaris connectats, també la guardarà a la classe aula. El motiu pel qual es fa d'aquesta manera és garantir que un usuari que entri a una aula on ja s'hagi fet servir la pissarra, i que per tant ja contingui informació rellevant, pugui rebre-ho i seguir l'explicació normalment.

El mateix passa en el cas de les diapositives. L'aula disposa d'una llista on es guarden tant bon punt arriben al servidor. això permet que l'alumne que entra nou a l'aula les pugui recuperar fàcilment. A més a més, també es guarda quina és la diapositiva que s'esta explicant en el moment de la connexió connexió per, de nou, enviar-li a l'alumne.

4.2 Client

Tecnologies utilitzades

Per al desenvolupament del client s'ha escollit combinar Unity com a plataforma de desenvolupament i C# com a llenguatge.

Per tal d'escollir la plataforma de desenvolupament, s'han valorat diferents opcions, entre elles: Unity, Unreal i Godot. Per prendre la decisió s'han valorat diferents aspectes com per exemple la manera de programar, els possibles llenguatges i el suport i la comunitat disponibles. Un cop fetes aquestes valoracions s'han extret les conclusions que es mostren a continuació.

Pel que fa a Unreal, tot i ser una molt bona plataforma, està molt més centrada en el disseny gràfic i el llenguatge de programació que ofereix és C++ amb diverses llibreries i macros, fet que fa que sigui complicat. L'alternativa a aquest llenguatge és utilitzar el blocs que ofereix com a format de programació més visual. En canvi, Unity està més centrat en la programació que hem après al llarg dels anys a la carrera i en llenguatges més familiars, això fa que que aquesta plataforma sigui molt més atractiva per a la gent que es dedica a programar.

Godot podria ser una molt bona opció, no obstant, al ser una plataforma més nova que la resta, disposa de moltes menys característiques i te una comunitat molt més petita. Això fa que no sigui tant viable com Unity.

Per tots aquests motius, s'ha decidit que la millor opció era Unity.

Pel que fa als possibles llenguatges de programació, Unity ens ofereix la possibilitat de programar en C# i JavaScript. A continuació es mostren les reflexions fetes per tal de decidir quin llenguatge seria utilitzat.

Obviant que la versió de JavaScript que ofereix Unity és una adaptació del llenguatge, el suport que ofereix Unity per aquest llenguatge no és total ja que no s'inclou en les noves versions. Per altre banda, C# si que disposa d'un suport complert, convertint-lo

així en una molt més bona opció. Addicionalment, és un llenguatge molt similar a Java, que és el que hem utilitzat més al llarg de la carrera i per tant és més fàcil adaptar-s'hi.

A més a més, JavaScript és un llenguatge que ofereix moltes formes diferents de fer una mateixa cosa i algunes d'elles no fan el que l'usuari espera, sobretot pels programadors inexperts o nous en la utilització d'aquest. Això fa que sigui lleugerament més complicat i que sigui menys recomanable.

L'editor que ha estat utilitzat per programar-ho és VisualStudio Code ja que és l'editor gratuït que te millor compatibilitat amb Unity a Sistemes Operatius Linux, que és l'utilitzat per el desenvolupament d'aquest projecte.

Desenvolupament

Per desenvolupar aquest client han estat utilitzades diferents escenes de Unity. Una escena és l'espai sobre el que es treballa i es programa a Unity. Són els conjunts d'objectes que formen les diferents pantalles que pot veure l'usuari. S'han utilitzat diverses escenes ja que el canviar d'una a l'altre garanteix resetejar els elements que hi apareixen i facilita la programació. En total disposem de quatre escenes:

- **Log In:** És l'escena que intenta realitzar una connexió amb el servidor i permet loguejar-se.
- **Choose character:** En aquesta escena l'usuari pot escollir quin serà el seu avatar i informar al servidor de la seva decisió.
- **Main Scene:** És l'escena que representa l'escola i, per tant, el servidor general.
- **Aula:** Aquesta escena representa cada una de les aules en concret i permet als usuaris fer totes les accions que fan referència a fer i assistir a classe.

El client està desenvolupat de forma asíncrona ja que la recepció i l'enviament no són bloquejants. Es fa d'aquesta manera per tal d'evitar que l'execució del client sigui aturada mentre espera la resposta del servidor. Per desenvolupar-ho, s'ha utilitzat com a referència l'exemple proporcionat per Microsoft[25].

Per tal de crear l'entorn gràfic i el personatge s'han utilitzat assets oferits per la comunitat de Unity. Per a fer l'aspecte de l'escola, s'ha fet servir l'*School assets*[26], l'*Street Lamps*[27], l'*Small town* [28] i finalment el *Free trees*[29]. Pel que fa al personatge, s'ha utilitzat el *Cute Male 3*[30].

La gestió de les respostes del servidor es tracta des d'una sola classe i es fa de forma asíncrona, tal com comentàvem anteriorment. Però en Unity, tot el que te a veure amb l'entorn gràfic s'ha de fer des del thread principal, fent que sigui un problema informar a l'entorn gràfic de quines son aquestes respostes per tal de que ell les gestioni. Per soluciona-ho s'ha utilitzat una *ConcurrentQueue* on s'afegeixen totes les accions necessàries per tal de que el fil principal les tregui i executi quan correspongui. Aquesta estructura és com una cua tradicional on la inserció i l'extracció són bloquejants de tal manera que no se'n poden fer dues alhora. La lògica seguida es pot veure en el pseudocodi que es mostra als algoritmes 1 i 2.

Per facilitar aquesta feina, en totes aquelles classes que contenen funcions que són afegides a la cua de la que parlàvem al paràgraf anterior, s'hi ha implementat una estructura similar al patró de disseny *Singleton*. Per fer això, les classes disposen d'un atribut estàtic anomenat *instance* on es guarden a elles mateixes. Això garanteix que tothom accedirà a la mateixa instància de la classe i facilitarà l'accés. S'ha obviat però la part del patró que fa que només hi pugui haver una instància de cada classe ja que Unity ja ho garanteix.

Algoritme 1: Procés d'encuament(Client)

```
while packet received do
    switch OpCode do
        case MOVEMENT do
            gm.instance.queue.Enqueue(lambda : gm.instance.player.move(...));
        end
        case MESSAGE do
            gm.instance.queue.Enqueue(lambda : aula.instance.AddMessage(...));
        end
        ...
    end
end
```

Algoritme 2: Procés de desencuament (GameManager)

```
while queue.Dequeue(action) do
    begin
        - player.move(...);
        - aula.instance.AddMessage(...);
        ...
    end
end
```

La càmera que permet veure el món està situada darrere del personatge i es mou amb ell. Això permet que l'usuari pugui veure bé el món que l'envolta. Per aquest punt s'ha valorat l'opció de posar la càmera des de diversos punts i inclinacions, com ara una vista superior o en primera persona. Finalment, després de provar i valorar totes les possibilitats es va escollir aquesta ja que era la que permetia una ubicació més natural en el món i una millor visualització de l'entorn. Ens podem fer una idea de com és aquesta perspectiva mitjançant la Figura 4.3.

A més a més, al llarg del desenvolupament s'han gestionat alguns casos extrems com els que es comentaran a continuació.

En primer lloc, la posició de la càmera respecte el personatge pot arribar a ser conflictiva ja que es poden donar casos en els que entre l'usuari i la càmera hi queda algun object. Això podria passar quan l'usuari s'apropa molt a un objecte i aleshores gira. No obstant, l'únic cas conflictiu és el cas en el que intervenen les parets fent que no es pugui veure res i dificultant la interacció amb el món. Per evitar això s'ha optat per ocultar les parets quan es donen aquests casos. Això permet que l'usuari segueixi tenint la visualització de l'espai i pugui ubicar-se i entengui on esta situat.

Figura 4.3: Exemple de la perspectiva de la camera.

Per tal de gestionar aquests casos, el que s'ha fet ha estat utilitzar RayTracing i mirar les interseccions. Cada un cert període de temps, es llençarà un raig des de la càmera fins al personatge. Si el raig intersecciona amb una paret, es comprovarà si aquesta ha estat amagada anteriorment. En cas negatiu, s'amagarà per tal de que l'usuari pugui veure el món correctament i es comprovarà si n'hi ha alguna altre d'amagada que s'hagi de fer visible. En cas de que la intersecció sigui amb el personatge, voldrà dir que no hi ha cap objecte, i per tant es faran visibles totes les parets amagades anteriorment. Es pot veure aquesta lògica al pseudocòdi 3 i el resultat a la Figura 4.4

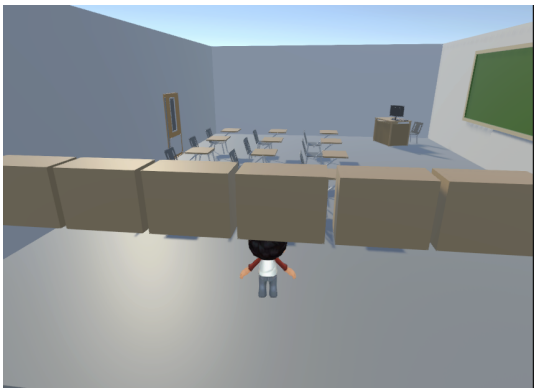


Figura 4.4: Resultat del RayTracing aplicat pel control de la intersecció amb parets.

Algoritme 3: Gestió d'intersecció amb parets mitjançant RayTracing

```
while true do
    if ray(camera, character) then
        if ray.hit = wall then
            if hit.object != lastHit then
                hit.object.invisible();
                if lastHit != null then
                    lastHit.visible();
                end
                lastHit = hit.object;
            end
        else
            if lastHit != null then
                lastHit.visible();
            end
            lastHit = null;
        end
    else
        if lastHit != null then
            lastHit.visible();
        end
        lastHit = null;
    end
end
```

Els altres casos extrems que s'han gestionat són els que tenen a veure amb els possibles errors de connexió amb els servidors.

Quan l'usuari intenta entrar a la plataforma, si el servidor general no s'està executant, aquesta connexió no serà possible i no es passarà mai de la pantalla inicial. Per evitar-ho, el client informa a l'usuari de que no ha estat possible la connexió i tanca el programa.

Per altre banda, quan un usuari intenta entrar a una aula però el servidor encara no està executant, apareix un missatge de que l'aula està tancada i l'usuari pot seguir interactuant amb l'escena principal. Per tal de poder dur a terme això, s'ha implementat el canvi de servidor de manera que abans de desconnectar-se del servidor principal, l'usuari s'assegura de que la connexió amb l'aula s'ha dut a terme correctament. Si aquest és el cas, l'usuari es desconnectarà del servidor general i es connectarà a l'aula.

En el plantejament del problema han estat mencionats alguns aspectes que s'havien de tenir en compte alhora de desenvolupar les eines de compartir i enviar pissarra i compartir veu. A continuació es discutiran els detalls de les solucions que s'hi han trobat.

La pissarra està composta de dues parts, per una banda tenim les diapositives i per l'altre els esbossos. Aquestes components, encara que a simple vista puguin semblar-ne una de sola, estan implementades de forma independent. Cada una es guarda en una imatge diferent de l'altre. El motiu pel qual s'ha fet d'aquesta manera és poder-les enviar per separat.

Mentre les diapositives s'envien només en el moment en el que son carregades i es guarden al servidor, la pissarra s'envia un cop cada un cert temps, sempre i quan s'estigui modificant. Fer-ho per separat ens permet optimitzar el procés el màxim possible ja que enviar una imatge amb només els esbossos és molt menys costos que enviar-la juntament amb la diapositiva i tots els bytes que això comporta.

No obstant, això també significa que un usuari que obri la pantalla quan les diapositives ja han estat penjades no les rebrà. Per solucionar-ho, quan un usuari obra la pissarra, demanarà al servidor si hi ha diapositives disponibles i, en cas de que n'hi hagi, seran enviades. Es seguirà la mateixa lògica pel cas de dibuixar a la pissarra, l'usuari ho demanarà i en cas de que en algun moment s'hi hagi escrit, s'enviaran els bytes corresponents.

Per tal d'enviar les diapositives i la pissarra, el que farem serà agafar la imatge que es mostra a la pissarra i codificar-la en format png. Podem veure el pseudocodi que representa aquestes funcions al pseudocodi 4.

Algoritme 4: Enviament de les diapositives i la pissarra

```
if drawing && time then
    board.toPNG();
    board.Send();
end
if uploadSlides then
    flies = filesSelected();
    for file in files do
        file.toPNG(); file.Send();
    end
end
```

Per tal d'implementar aquestes funcionalitats s'han utilitzat alguns recursos publicats per la comunitat d'usuaris de Unity. Per la pissarra s'ha utilitzat com a referència un dels assets gratuïts que ofereix Unity, anomenat *Free Draw* [31] modificant els seus scripts per tal d'adaptar-los a aquest projecte. Pel que fa a la càrrega de fitxers, s'ha utilitzat el repositòri de *GitHub* anomenat *Unity Standalone File Browser* [32].

La freqüència d'enviament de les diapositives ha estat escollida després de fer diferents proves. En aquestes proves s'ha demostrat que el millor interval d'enviament era cada 1 segon. Aquest interval ens garanteix que l'alumne rep els canvis de la forma més fluida possible i que el sistema no queda sobresaturat.

Finalment, pel que fa referència a l'enviament de veu, s'ha fet també cada un segon ja que també era la relació temps/enviament que garantia unes millors condicions tant per l'usuari com pel sistema. A banda d'això, s'ha preparat el sistema de manera que quan l'usuari rebi un paquet d'àudio, no el comenci a reproduir directament sinó que esperi a que arribi el segon. Això és per tal d'evitar també que hi hagi talls entre l'un

i l'altre ja que quan un s'acabi de reproduir, sempre tindrem el següent esperant per fer-ho. Podem veure el pseudocòdi que descriu aquest procés a l'algoritme 5.

Algoritme 5: Acumulació de fragments d'audio

```
if audio && time then
    audioList.Enqueue();
    if audioList.Count > 1 && !reproducing then
        reproducing = true;
        reproduce();
    end
end
Function reproduce() is
    while audioList.Dequeue() do
        audio.Play();
    end
end
```

Per defecte, aquesta opció està desactivada i no és fins que el professor l'activa manualment que es comença a enregistrar i a enviar.

La pissarra també disposa d'un espai on els alumnes puguin escriure per tal de consultar dubtes i conversar amb els altres alumnes o amb el professor.

En el diagrama que es mostra a la Figura 4.5 podem veure en quin ordre s'instancien les diferents classes que componen el client, separades per les escenes en les que han estat creades. Es pot observar que hi ha algunes escenes que utilitzen classes que han estat instanciades en d'altres escenes. Això és degut a que es fa una crida *don't destroy on load* que permet que alguna informació sigui conservada entre les diferents escenes.

4.3 Base de dades

Tecnologies utilitzades

Pel que fa a la base de dades, finalment s'ha optat per utilitzar les tecnologies que s'expliquen a continuació.

Com a llenguatge s'ha utilitzat SQL. Aquest és un llenguatge molt popular i utilitzat, te una gran adaptabilitat i funcionava bé per el tipus de dades que es necessitaven guardar, ja que davant de l'opció de guardar-les de forma relacional o bé en documents, la primera era més pràctica i òptima. A més a més, és el llenguatge que ens han ensenyat a la carrera.

El sistema gestor de bases de dades utilitzat ha estat MySQL. A banda de que aquest sistema és el que se'ns ha ensenyat a la carrera, també és molt fàcil de configurar i està disponible per a totes les distribucions de sistemes operatius.

Desenvolupament

Per realitzar la connexió des del servidor, s'ha utilitzat l'API *jdbc* de Java, que permet accedir a bases de dades a través d'una interfície comuna.

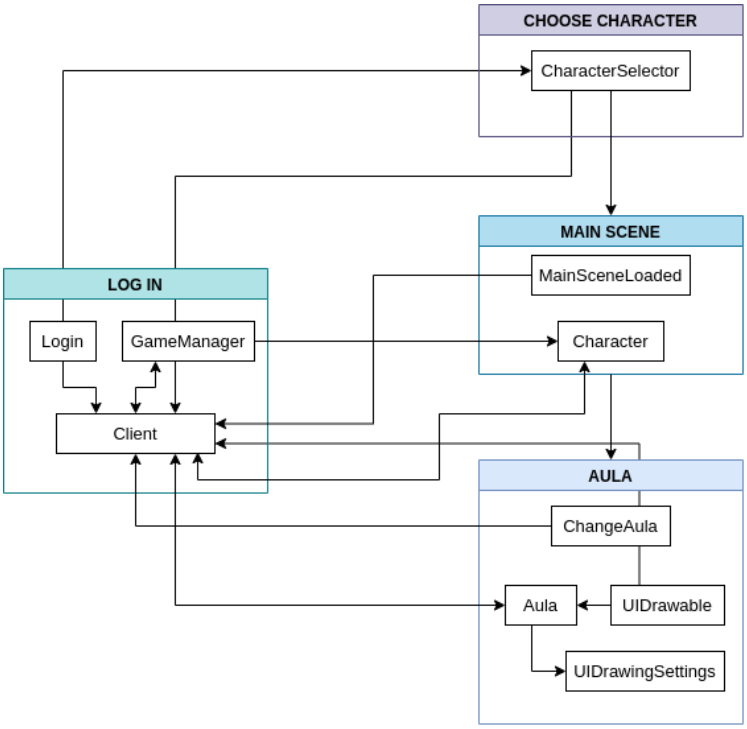


Figura 4.5: Diagrama representatiu de les classes i escenes del client. Les caixes grans que contenen títol representen les diferents escenes que podem trobar, mentre que les petites representen les diferents classes disponibles. Es mostra quines classes s'instancien i a quines escenes i també com es relacionen i es criden entre elles.

A l'hora de dissenyar la base de dades s'ha fet de manera que aquesta estigui normalitzada i compleixi amb les 3 formes normals:

- Una taula està en 1FN si els seus atributs son atòmics(no descomposables, no agregables i no repetitiu).
- Una taula està en 2FN si esta en 1FN i tots els atributs no clau depenen de la clau primària.
- Una taula està en 3FN si esta en 2FN i cap atribut no clau depèn d'un altre atribut no clau.

És per això que actualment esta composta per tres taules, que son les necessàries per gestionar els usuaris i contrasenyes i els seus rols. Podem veure les taules a la Figura 4.6

- users(id, name, password, salt)
- roles(id, role)

– users_roles(id_user, id_role)

Per seguretat, les contrasenyes dels usuaris es guarden seguint un procés de hash. Els més comuns són *md5* i *SHA*. Com que aquest primer fa molt de temps que existeix i és molt fàcil de revertir, ja no és una opció segura i és per això que s'ha decidit utilitzar *SHA*.

Aquest mètode passa la contrasenya per un doble procés de codificació. La primera vegada quan aquesta surt del Client i la segona un cop la rep. En aquesta segona codificació, se suma un valor random únic, anomenat *salt*, per a cada usuari. El motiu pel qual es fa això és que, si el paquet és interceptat durant el procés d'enviament, es podria fer servir igualment aquesta informació per realitzar actes maliciosos.

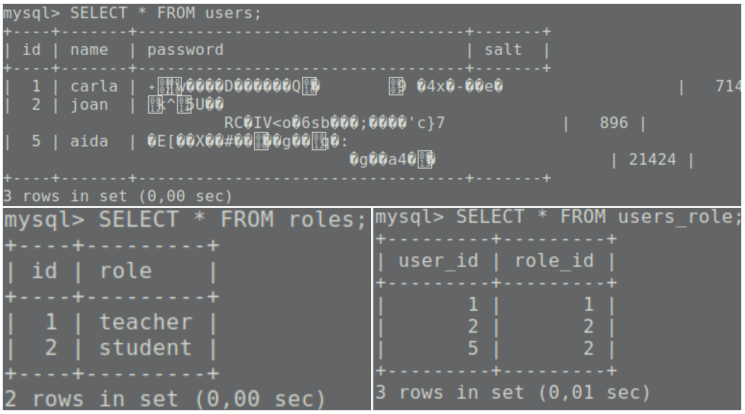


Figura 4.6: Estat actual de les taules a la base de dades.

4.4 Protocol

Per tal de realitzar la comunicació correctament ha estat necessari definir un protocol. Un protocol de comunicació és una definició formal dels formats que han de tenir els missatges per poder ser intercanviats entre els diferents equips.

El protocol implementat en aquest projecte és binari, aquest tipus de protocols estan orientats a les estructures de dades. L'alternativa seria fer un protocol de text, que estan orientats a *strings*. Fer-ho d'aquesta manera ens garanteix que la comunicació és el màxim òptima possible.

No obstant, que el protocol sigui binari implica que el contingut de tots els paquets s'haurà de traduir abans de l'enviament del seu format a binari i, després de la recepció, de binari al format original. Al parlar de traduir, ens referim a escriure els bytes en el paquet amb un ordre concret i llegir-los segons aquest per tal d'assegurar-nos de que tots els equips ho fan de la mateixa manera. En aquest cas s'ha utilitzat *Little Endian* ja que és el que acostumen a utilitzar els ordinadors moderns i a més a més és el més comú en comunicacions per la xarxa. Per resoldre aquest problema s'han desenvolupat totes les funcions corresponents que permeten aquesta traducció tant a la banda del client com a la banda del servidor. És important comentar que, en el cas de les cadenes

de caràcters, a part d'escriure els bytes que les componen, també s'escriu un enter que representa la mida d'aquesta per tal d'assegurar-nos de que es llegirà tota.

El protocol consta d'una capçalera de 5 bytes seguida d'un cos de mida variable. En els 5 bytes de capçalera hi podem trobar 1 byte per l'*OpCode* (el codi d'operació) i un enter que indicarà la mida del cos (Figura 4.7).

Figura 4.7: Estructura del protocol.

Per tal de fer el codi més entenedor i fàcil de mantenir, tant a la banda del client com a la banda del servidor, s'han definit un conjunt d'atributs que, donat un text que representa el nom del paquet, el converteixen al byte corresponent i viceversa. Això ens permet poder identificar fàcilment quina part del codi gestiona cada paquet.

El fet de que la capçalera contingui la mida del cos ens ajuda a comprovar si ens ha arribat tot el paquet i també ens permet realitzar parts del sistema de forma asíncrona. A banda d'això, també facilita l'ampliació del protocol ja que, quan es vulgui afegir un nou paquet, només s'haurà de definir un nou OpCode i una funció que el pugui escriure i llegir.

Els paquets que formen el protocol són els que es poden observar a la Taula 4.1 i Taula 4.2, que representen els paquets que pot enviar el client i el servidor, respectivament.

En aquest protocol disposem de diferents tipus de paquets, per una banda tenim tots els que ens permeten la connexió i la desconnexió del sistema. Aquests inclouen des de fer un log-in i comprovar-lo fins a informar a tots els usuaris de quan algú s'està connectant i desconnectant. Per altre banda, tenim els paquets que controlen el moviment, el canvi d'escenes i l'actualització dels usuaris connectats. Finalment tenim els paquets que gestionen les opcions disponibles a la pissarra i s'encarreguen d'enviar la informació necessària a tots els usuaris per tal d'obtenir la visualització completa.

Tenint en compte els paquets que s'han explicat i totes les accions disponibles, a continuació es poden veure els diagrames d'interacció entre el Client i el Servidor.

A la Figura 4.8 podem veure el flux que hi ha entre el client i el servidor a l'hora de fer una connexió. Com es pot veure amb la primera fletxa, és el client el que s'encarrega de demanar la connexió al servidor i, un cop aquesta s'ha establert, el servidor li proporciona al client un identificador. Quan aquest procés s'ha fet, l'usuari pot començar a interactuar amb la plataforma i entrar dins el món virtual mitjançant el log-in i l'elecció d'avatar. Un cop l'usuari ha enviat la informació referent al seu avatar (paquet *DATA*), el client informa a la resta d'usuaris de l'escola de que un nou usuari s'ha connectat i, per tant, s'ha d'instanciar en els seus mons.

Taula 4.1: Llista de paquets que pot enviar el client

0: LogIn				
OpCode	**Size**	**Uname**	**Password**	
1B	4B	XB + 4B	XB + 4B	
1: Data				
OpCode	**Size**	**Uname**	**prefab**	
1B	4B	XB + 4B	4B	
2: Moving				
OpCode	**Size**	**Position**		
1B	4B	12B		
3: Leaving				
OpCode	**Size**	**Connection (1/0)**		
1B	4B	1B		
4: Message				
OpCode	**Size**	**Receiver**	**Sender**	**Message**
1B	4B	4B	4B	XB + 4B
5: Join				
OpCode	**Size**	**Uname**	**prefab**	
1B	4B	XB + 4B	4B	
6: Board				
OpCode	**Size**	**Board**		
1B	4B	XB		
7: Ask Board				
OpCode	**Size**			
1B	4B			
8: Existing				
OpCode	**Size**			
1B	4B			
9: Slide				
OpCode	**Size**	**Slide**		
1B	4B	XB		
10: Slide Index				
OpCode	**Size**	**Index**		
1B	4B	4B		
11: Empty Slide				
OpCode	**Size**			
1B	4B			
12: Voice				
OpCode	**Size**	**Voice**		
1B	4B	XB		
13: End Voice				
OpCode	**Size**			
1B	4B			

Taula 4.2: Llista de paquets que pot enviar el servidor

0: LogIn			
OpCode	**Size**	**Correct**	**Role**
1B	4B	1B	4B
1: ID			
OpCode	**Size**	**ID**	
1B	4B	4B	
2: Connection			
OpCode	**Size**	**ID**	**Connection (1/0)**
1B	4B	4B	1B
3: Movement			
OpCode	**Size**	**ID**	**Position**
1B	4B	4B	12B
4: Change			
OpCode	**Size**	**IP**	**Port**
1B	4B	XB + 4B	4B
5: Message			
OpCode	**Size**	**String**	
1B	4B	XB + 4B	
6: Board			
OpCode	**Size**	**Board**	
1B	4B	XB	
7: Slide			
OpCode	**Size**	**Slide**	
1B	4B	XB	
8: Slide Index			
OpCode	**Size**	**Index**	
1B	4B	4B	
9: Empty Slide			
OpCode	**Size**		
1B	4B		
10: Voice			
OpCode	**Size**	**Voice**	
1B	4B	XB	
11: End Voice			
OpCode	**Size**		
1B	4B		

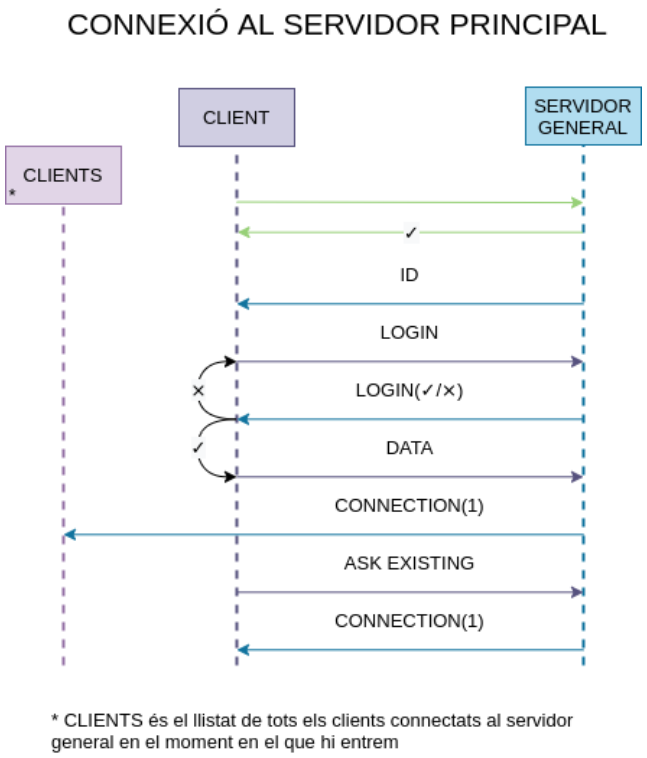


Figura 4.8: Diagrama d'interacció pel cas de connectar-se al servidor principal.

Un cop el client s'ha connectat al servidor general ja pot començar a interactuar amb el món. Tal i com es pot veure en el diagrama de la Figura 4.9, en aquest espai hi ha poques interaccions a fer ja que està pensat per ser un espai de pas. L'usuari únicament podrà moure's pel món i desconnectar-se.

El moviment es registra a partir de *clicks* a la pantalla. Cada vegada que l'usuari en faci un, s'enviarà la posició d'aquest i, un cop el servidor la rebi, després de fer les comprovacions corresponents, la reenviarà a tots els usuaris per tal de que actualitzin la posició d'aquesta persona en el món. Això també l'inclou a ell mateix.

En el diagrama de la Figura 4.10 es mostra el flux a l'hora de canviar de servidor. Com es pot veure, quan el client es mou a un punt proper a la porta, el servidor l'informa de que ha de realitzar un canvi de servidor. Quan aquest ho rep, intenta connectar-se a l'aula abans de desconnectar-se del servidor. No és fins que ha rebut que la connexió ha estat acceptada i realitzada amb èxit que avisa al servidor general de que deixa l'escena i es connecta definitivament a l'aula, on es segueix el mateix procediment de connexió al servidor.

A la Figura 4.11 podem veure el diagrama que explica com és l'intercanvi de paquets entre el client i el servidor en totes les possibles interaccions comunes per a alumnes i professors dins de l'aula. Es pot veure quina és la lògica que se segueix quan una persona obra la pissarra. Al obrir-la, el client enviarà una petició al servidor on demanarà,

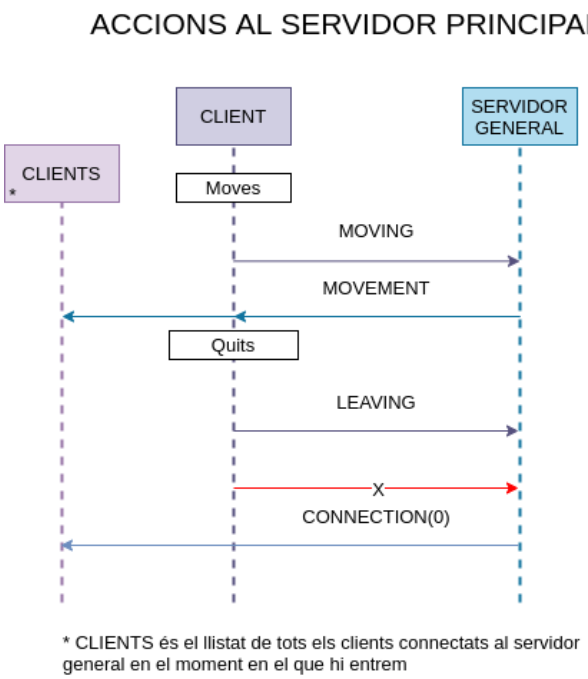


Figura 4.9: Diagrama d'interacció per les possibles accions al servidor principal.

si existeixen, quins son els continguts actuals a mostrar i aquest contestarà únicament si hi ha algun tipus de contingut. Podem veure també que els usuaris poden enviar-se missatges mitjançant un xat.

En el diagrama que es mostra a la Figura 4.12 es pot veure l'intercanvi de paquets que succeeix entre el client i el servidor en les accions exclusives del professorat. En aquest diagrama es pot veure què passa quan un professor carrega una sèrie de diapositives. Tant bon punt s'han carregat, el client les envia al servidor de tal manera que aquest se les guardi i les re-enviï a tots els clients connectats a l'aula en aquell moment. Això permet que quan el professor vulgui canviar de diapositiva, únicament hagi d'informar de la posició de la diapositiva que es vol mostrar. Pel que fa a la veu, tant bon punt el professor activa la opció, el client començarà a gravar i enviarà petits trossos cada un cert temps.

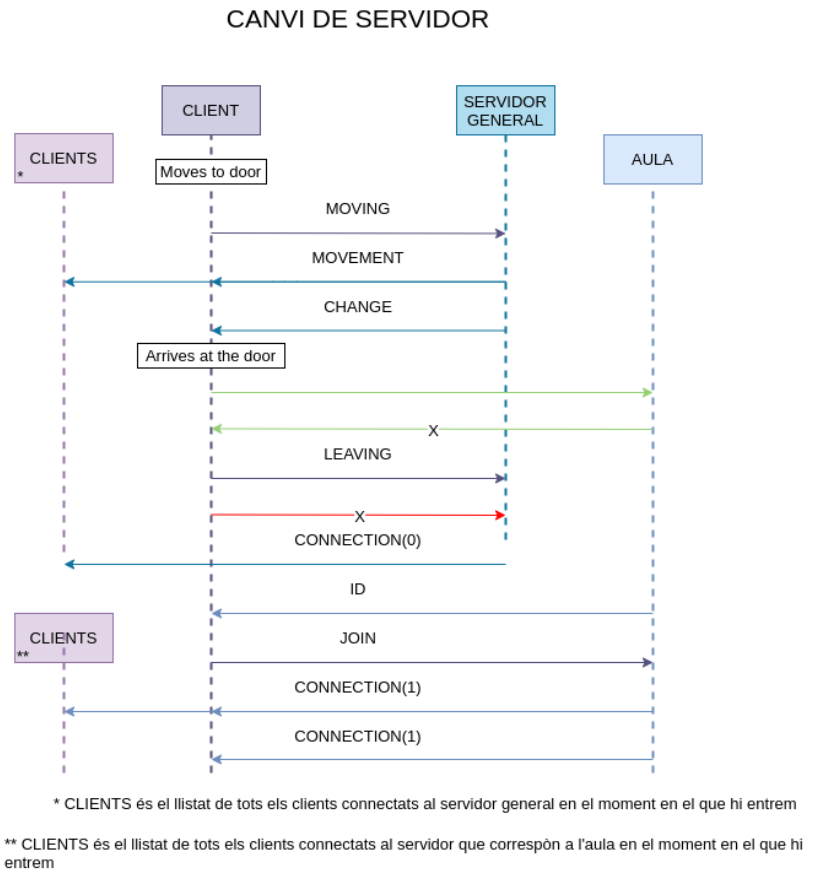


Figura 4.10: Diagrama d'interacció per quan es canvia de servidor.

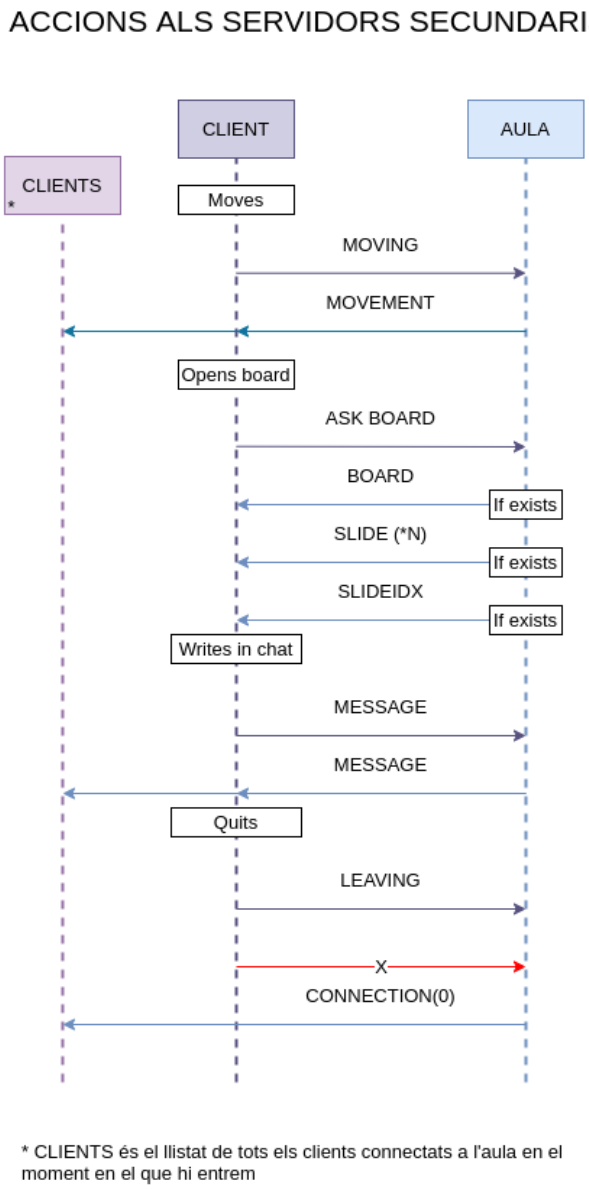


Figura 4.11: Diagrama d'interacció per les accions que poden fer dins l'aula tots els usuaris.

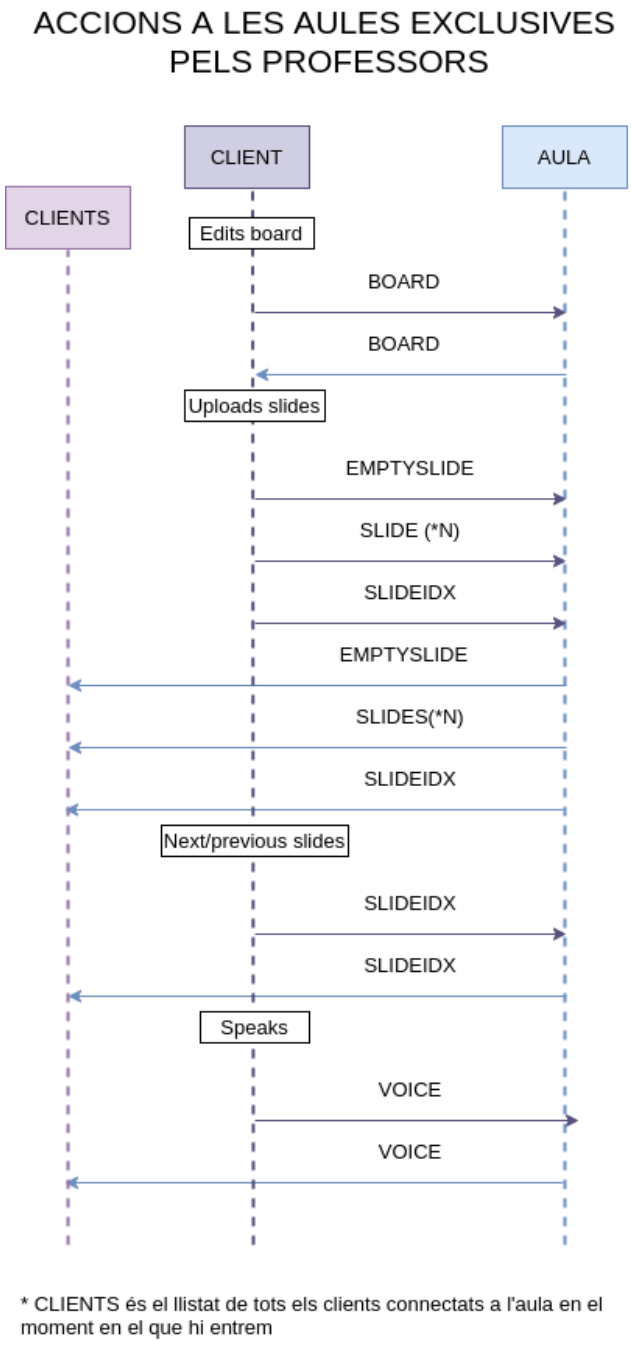


Figura 4.12: Diagrama d'interacció per les accions que només els professors poden fer dins l'aula.

5. Simulacions i resultats

En aquest capítol s'explicarà quin ha estat el resultat del projecte i quines han estat les proves realitzades per tal de comprovar-ne les seves funcionalitats i la seva usabilitat.

5.1 Resultats

Com s'ha comentat anteriorment, la plataforma disposa de quatre escenes amb les que els usuaris poden interaccionar.

Login

En aquesta pantalla les accions disponibles són únicament connectar-se al servidor, tal i com es pot apreciar a la Figura 5.1. En cas de no ser possible, es mostra un missatge informatiu a l'usuari que, un cop s'accepta, tanca la plataforma 5.2. També és mostra un missatge quan l'usuari o la contrassenya son erronis, com es pot veure a la Figura 5.3.

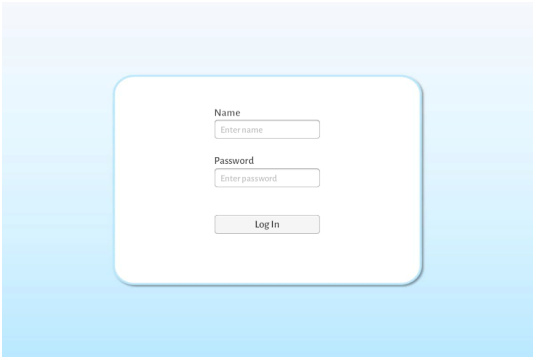


Figura 5.1: Pantalla de LogIn.

Choose character

En aquesta escena l'usuari pot escollir quin serà el seu avatar. Mitjançant les fletxes, que poden ser accionades amb el ratolí o amb el teclat, l'usuari pot veure les diferens opcions i seleccionar la desitjada 5.4.

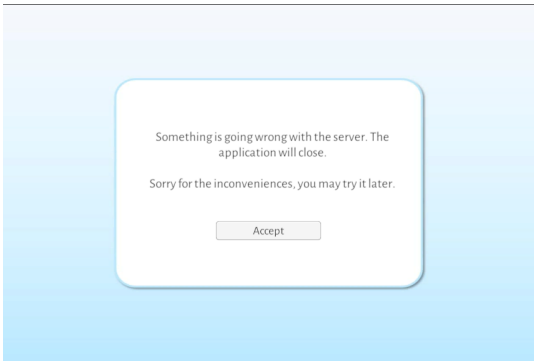


Figura 5.2: Pantalla d'error al connectar al servidor general.

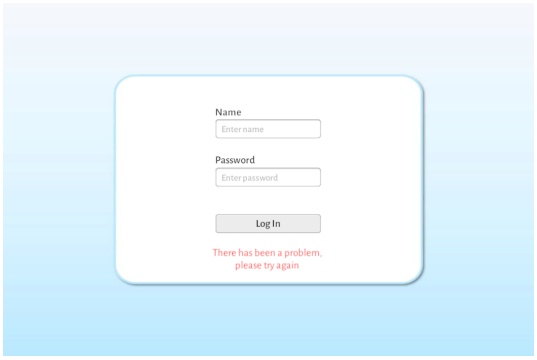


Figura 5.3: Pantalla d'error a l'introduïr les dades.

Figura 5.4: Pantalla per escollir l'avatar i les seves opcions.

Main scene

Aquesta escena representa l'escola. És l'escena de pas que permet entrar a l'aula. Es pot veure el seu aspecte a la Figura 5.5. Per entrar a l'aula el que haurem de fer és fer un *click* sobre la porta.

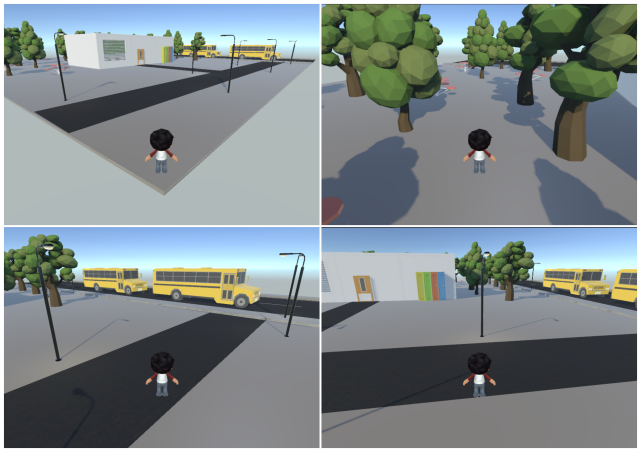


Figura 5.5: Visualitzacions de l'escena principal des de diversos punts de vista.

Podria passar que l'aula no estigues oberta quan s'hi vol entrar, en aquest cas s'informa a l'usuari per tal de que sàpiga que encara no pot, com es mostra a la Figura 5.6.

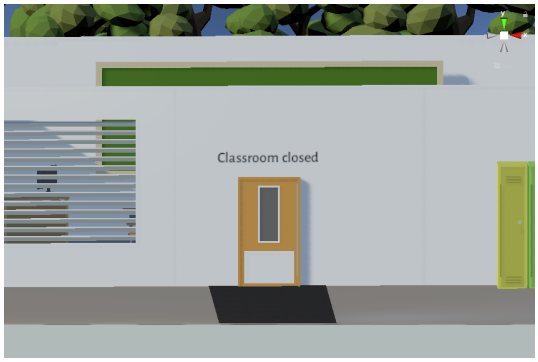


Figura 5.6: Missatge d'error a l'entrar a l'aula quan no està disponible.

Aula

Aquesta escena te dues possibles visualitzacions. El cas en el que s'ha entrat a l'aula i s'interacciona amb ella 5.7 i el cas en el que s'ha obert la pissarra 5.8.

La pissarra pot tenir diferents visualitzacions, la versió del professor i la versió de l'alumne. En aquesta primera hi apareixen alguns botons que permeten realitzar les diferents funcionalitats disponibles 5.9.

Figura 5.7: Visuaització de l'aula a l'entrar a l'escena.

Figura 5.8: Aspecte de la pissarra.

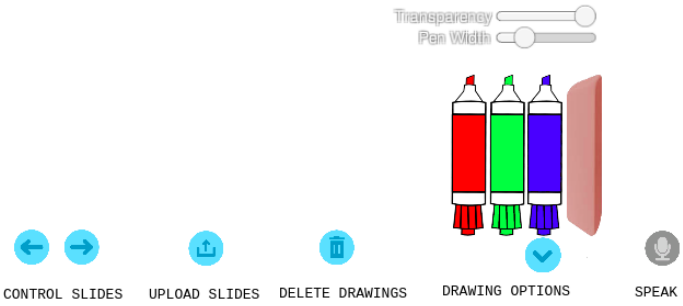


Figura 5.9: Accions disponibles per al professor a la pissarra.

Disponibilitat del codi

El projecte s'ha desenvolupat en tot moment utilitzant GitHub com a eina de control de versions. El motiu d'utilitzar aquesta plataforma no ha estat només poder tenir còpies de seguretat disponibles. També s'ha fet d'aquesta manera per tal de poder-la convertir en una plataforma de codi lliure i gratuïta per a tots aquells usuaris als que els pugui interessar utilitzar-la i ampliar-la.

Taula 5.1: Taula d'usuaris

Alumnes		**Professors**	
Usuari	Contrassenya	Usuari	Contrassenya
joan	joan	carla	carla
aida	snoopy135		

Els repositòris de GitHub on podem trobar el codi es llisten a continuació:

- Client: https://github.com/CarlaMontserrat/ClientTFG.
- Servidor: https://github.com/CarlaMontserrat/ServidorTFG

Aquests repositòris es faran públics a partir de la data d'entrega del treball.

També es pot executar la versió actual descarregant-ho seguint el manual tècnic i utilitzant els usuaris mostrats a la Taula 5.1.

5.2 Tests

Un cop realitzat aquest projecte, s'han dut a terme algunes proves per tal de comprovar-ne el funcionament i s'ha demanat a usuaris de diferents perfils que l'utilitzessin per tal de recollir-ne les experiències i millorar-lo.

Pel que fa a la seva funcionalitat, s'ha provat d'executar-lo de diferents maneres i en diferents condicions.

En primer lloc, es va comprovar la seva funcionalitat a l'executar diferents clients des d'una xarxa local, és a dir, des del mateix ordinador. El sistema va resultar en un funcionament correcte i sense cap mena de delay.

Seguidament es va provar de fer el mateix però en una xarxa wifi. El resultat va ser el mateix, funcionalitat correcte i sense delays.

Finalment es va provar d'ubicar el codi del servidor en un servidor accessible per tota la xarxa ubicat a França, i els clients executats des de diferents xarxes wifi. La funcionalitat va resultar correcte però en aquest cas si que s'apreciava una mica més de retards en la comunicació però, en cap cas eren significatius.

Pel que fa a les proves amb usuaris, la majoria d'ells se'n van sortir molt bé i van poder provar totes les funcionalitats sense ajuda i amb un temps raonable.

En alguns casos els usuaris van trobar complicat descobrir com s'havien de moure l'avatar i els sorgia intuïtivament fer-ho mitjançant les fletxes del teclat. Als usuaris que els va passar això eren persones poc acostumades a utilitzar plataformes virtuals o bé usuaris de jocs que es controlen mitjançant el teclat.

A altres usuaris poc acostumats a les noves tecnologies també els va costar entendre com s'havia d'entrar a la classe o bé com s'havia d'accedir a la pissarra. Però tots ho van acabar deduint sense ajuda.

Per tal de solucionar aquest problema, en versions futures es procurarà incloure una guia de les funcionalitats on s'expliqui com es realitzen cada una d'elles.

A les primeres proves també es van detectar problemes amb la rotació del personatge i la posició de la càmera. Aquestes no estaven del tot ben aconseguides i marejaven una mica als usuaris. Es va tenir en compte i es va canviar per tal d'evitar-ho.

6. Conclusions i treball futur

En aquest apartat seran avaluats els objectius plantejats a l'inici del projecte i el seu assoliment. També es plantejaran les possibles ampliacions que podrien afegir-s'hi en un futur.

6.1 Conclusions

A l'inci d'aquest projecte es van plantejar uns objectius separats en dos blocs així com unes tasques i uns requisits per tal d'assolir-los. Un cop desenvolupat el projecte, l'opció més lògica és avaluar-los també contemplant aquesta divisió.

En primer lloc es parlava dels objectius personals, que estaven motivats per la voluntat d'aprendre noves tecnologíes i posar a prova tant les capacitats personals com els aprenentatges adquirits al llarg de la carrera. També per la necessitat de saber si seria capaç de desenvolupar un projecte des de 0 amb tot el que això comporta.

Un cop acabat al projecte i amb la perspectiva que això aporta, es pot concloure que aquest bloc ha estat completat amb èxit. No només he pogut aconseguir realitzar tot aquest projecte des del seu plantejament fins a la seva avaluació sinó que també m'ha permès posar en pràctica coneixements adquirits al llarg de tota la carrera i adquirir-ne de nous per tal d'ampliar-los. S'ha pogut aplicar coneixements bàsics de Disseny de Software, com la captura de requisits potencials i els patrons de disseny, aplicats en un cas real. He pogut ampliar les nocions d'arquitectures de comunicació explicades a Software Distribuït així com aplicar els coneixements de desenvolupament de protocols. També m'ha permès aplicar, ampliar i entendre millor alguns coneixements sobre interseccions amb l'escena mitjançant RayTracing, explicats a l'assignatura de Gràfics i Visualització de dades.

Per altre banda, els objectius centrats en els usuaris estaven motivats per la necessitat de millorar l'experiència d'alumnes i professors a l'hora de fer classes en línia. Però sobretot, millorar l'experiència i la qualitat en situacions tant complicades com les viscudes recentment, on l'estat d'ànim de la gent i la poca capacitat de captivar als alumnes ha fet dels estudis una tasca més difícil de l'habitual.

En aquesta línia de desenvolupament, es pretenien desenvolupar les bases d'una eina que permetés un realisme més complet del que disposen les plataformes actuals, que estan basades únicament en la connexió i no en la simulació d'un món. El producte final ha resultat en una plataforma que permet fer-se a la idea d'estar en un entorn es-

colar molt més familiar, encara que no permeti la immersió completa degut als objectes representats en forma de dibuix.

Tenint en compte tot això, podem considerar que aquesta segona línia d'objectius ha estat assolida correctament, doncs els usuaris poden realitzar totes les accions plantejades i d'una manera més senzilla i captivadora.

Després d'analitzar tots els objectius i el seu assoliment, podem concloure que el projecte ha estat realitzat i acabat amb èxit.

6.2 Treball futur

Al llarg del document, s'ha comentat diverses vegades que aquesta no és una versió final del producte i que està oberta a possibles millores futures. En aquest apartat s'anomenaran i es detallaran breument aquestes propostes.

En les primeres millores es podrien incorporar els requisits secundaris que es van definir abans de començar el projecte i que no s'han pogut desenvolupar en aquesta versió.

En aquest cas es prioritzaria implementar el Calendari i Tauler ja que són característiques que poden aportar una gran millora al projecte. Permetrien que els alumnes tinguessin un lloc on consultar el material explicat a classe i també un lloc on consultar les dates d'entrega.

Per implementar-los, seria necessari ampliar, per la part del client, les funcionalitats disponibles dins de l'aula, per tal de poder accedir als continguts. Per la part del servidor, s'hauria de modificar la base de dades per guardar tota la informació. Finalment, també s'hauria d'ampliar el protocol afegint-hi paquets de forma que tots els usuaris puguin accedir-hi.

A part d'afegir-hi aquestes funcionalitats, també es podrien aplicar les millores de les quals s'ha parlat al llarg del document. Aquestes millores inclouen el desenvolupament de la seguretat i el canvi de protocol en el cas de l'enviament de veu.

En termes de seguretat, s'hauria de procurar que ningú pugues accedir ni modificar la informació que contenen els paquets que s'envien el client i el servidor.

Pel que fa a l'enviament i la recepció de veu, desenvolupar l'ús protocol UDP. En termes generals, UDP és un millor protocol a utilitzar en aquests casos ja que no implementa ni el control ni la confirmació de flux, deixant la possibilitat de que els paquets es puguin perdre o avançar-se els uns als altres. Aquesta característica és important en casos com aquest ja que és preferible no sentir un tros de la conversa abans que rebre-la amb molt de retard i no poder seguir el fil correctament.

A. Manual tècnic

A.1 Versions de Software

Els requerimets del sistema per tal d'executar el programa són el següents:

Sistema operatiu	Windows 7,SP1 o superior Ubuntu 16.04 o superior
CPU	SSE2 instruction set support
GPU	Targeta gràfica amb DX10 (shader model 4.0) capabilities

A.2 Com instal·lar

Per a utilitzar la plataforma, només és necessari accedir a l'enllaç que es pot veure a continuació i descarregar-se la carpeta corresponent al sistema operatiu.

https://drive.google.com/drive/folders/1LZiwJLqrLrqfwvf14_kPC_n7NyUvWP40?usp=sharing

Un cop descarregada la carpeta i extret el zip, en el cas de Linux, serà necessari donar permisos a l'executable, anomenat Client-TFG-CarlaMontserrat.x86_64. Això es pot fer mitjançant la terminal i executant la comanda *chmod 777 Client-TFG-CarlaMontserrat.x86*$_6$*4desdelmateixdirectorionestubicatel fitxer*.

En el cas cas de Windows, n'hi hauarà prou amb executar l'arxiu Client-TFG-CarlaMontserrat.exe.

És important descarregar-se tota la carpeta i no moure els arxius de les seves carpetes, d'altre banda podria no executar-se correctament.

Bibliografia

[1] David E. Borth. Videophone. URL https://www.britannica.com/technology/videophone. Accessed on 05.06.2021.

[2] A brief history of video conferencing: From the beginning to full commercial use, 2019. URL https://habr.com/en/post/465459/. Accessed on 19.06.2021.

[3] Mechanical television - bell labs two-way television. URL http://www.earlytelevision.org/bell_labs.html. Accessed on 19.06.2021.

[4] Paul J Choi, Rod J Oskouian, and R Shane Tubbs. Telesurgery: past, present, and future. Cureus, 10(5), 2018.

[5] Charlotte Trueman. Pandemic leads to surge in video conferencing app downloads, 2020. URL https://www.computerworld.com/article/3535800/pandemic-leads-to-surge-in-video-conferencing-app-downloads.html. Accessed on 05.06.2021.

[6] C González. Gamificación en el aula: ludificando espacios de enseñanza-aprendizaje presenciales y espacios virtuales. Researchgate. net, pages 1–22, 2019.

[7] The education district, . URL https://www.theeducationdistrict.com/es/. Accessed on 15.02.2021.

[8] The education district - startupexplore, . URL https://startupxplore.com/es/startups/the-education-district. Accessed on 15.02.2021.

[9] Viajes educativos virtuales con guías reales, 2015. URL https://www.theeducationdistrict.com/es/educacion/viajes-educativos-virtuales. Accessed on 18.06.2021.

[10] Jonathan Kaplan and Nicole Yankelovich. Open wonderland: An extensible virtual world architecture. IEEE Internet Computing, 15(5):38–45, 2011.

[11] Openwonderland. URL http://openwonderland.org/. Accessed on 15.02.2021.

[12] Surinder Kahai. Wonderland: A tool for online collaboration, 2008. URL https://www.leadingvirtually.com/wonderland-a-tool-for-online-collaboration/. Accessed on 15.02.2021.

[13] *David Parsons and Rosemary Stockdale. Cloud as context: Virtual world learning with open wonderland. In* Proceedings of the 9th World Conference on Mobile and Contextual Learning, Malta, *pages 123–130, 2010.*

[14] *Nicole Yankelovich. Learning the basics of open wonderland (tutorial). URL https://sites.google.com/site/openwonderland/tutorials/learning-the-basics-tutorial. Accessed on 15.02.2021.*

[15] *Create virtual experiences. URL https://www.lindenlab.com/. Accessed on 18.06.2021.*

[16] *Secondlife, 2021. URL https://ca.wikipedia.org/wiki/Second_Life. Accessed on 15.02.2021.*

[17] *Kurt Winkelmann, Wendy Keeney-Kennicutt, Debra Fowler, and Maria Macik. Development, implementation, and assessment of general chemistry lab experiments performed in the virtual world of second life.* Journal of Chemical Education, *94 (7):849–858, 2017.*

[18] *Nadezhda Maksimenko, Anzhelika Okolzina, Anna Vlasova, Chantal Tracey, and Mikhail Kurushkin. Introducing atomic structure to first-year undergraduate chemistry students with an immersive virtual reality experience, 2021.*

[19] *Que es opensimulator?, 2012. URL http://opensimulator.org/wiki/ParaQueSirve. Accessed on 15.02.2021.*

[20] *Opensimulator, 2020. URL http://opensimulator.org/wiki/Main_Page. Accessed on 15.02.2021.*

[21] *Kohei Arai and Anik Nur Handayani. E-learning system utilizing learners' characteristics recognized through learning processes with open simulator.* International Journal of Advanced Research in Artificial Intelligence, *4(4):8–12, 2013.*

[22] *Ross Brown, Rune Rasmussen, Ian Baldwin, and Peta Wyeth. Design and implementation of a virtual world training simulation of icu first hour handover processes.* Australian Critical Care, *25(3):178–187, 2012.*

[23] *Opensimulator, 2021. URL https://es.wikipedia.org/wiki/OpenSimulator. Accessed on 15.02.2021.*

[24] *Sheri Espinoza. Infographic: Comparing video conferencing apps, 2020. URL https://www.uhclthesignal.com/wordpress/2020/05/04/infographic-comparing-video-conferencing-apps/. Accessed on 18.06.2021.*

[25] *Asynchronous client socket example, 2017. URL https://docs.microsoft.com/en-us/dotnet/framework/network-programming/asynchronous-client-socket-example. Accessed on 17.03.2021.*

[26] *Jarst. School asset, 2021. URL https://assetstore.unity.com/packages/3d/environments/school-assets-146253. AssetStore, Accessed on 18.05.2021.*

[27] *SpaceZeta. Street lamps, 2020. URL https://assetstore.unity.com/packages/3d/props/exterior/street-lamps-165658. AssetStore, Accessed on 16.06.2021.*

[28] MultiFlagStudios. Small town america - streets - free, 2016. URL https://assetstore.unity.com/packages/3d/small-town-america-streets-free-59759. AssetStore, Accessed on 16.06.2021.

[29] Ada$_k$ing. Freetrees, 2017. URL. AssetStore, Accessed on 16.06.2021.

TurkCheeps. Cute male3, 201. URL https://assetstore.unity.com/packages/3d/characters/cute-male-3-18090.

Foolish Mortals. Free draw - simple drawing on sprites/2d textures, 2019. URL https://assetstore.unity.com/packages/tools/painting/free-draw-simple-drawing-on-sprites-2d-textur content. AssetStore, Accessed on 20.03.2021.

Unity standalone file browser. URL https://github.com/gkngkc/UnityStandaloneFileBrowser.