UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

ESCOLA TÉCNICA SUPERIOR DE ENXEÑARÍA

Desenvolvemento dun entorno interactivo de baixo custo para a animación de personaxes utilizando dispositivos de captura de movemento sen marcas

MEMORIA
TRABALLO FIN DE GRAO

Presentada por:
Roberto Noya Noya

Dirixida por:
Paulo Félix Lamas
María José Carreira Nouche

Santiago de Compostela, Xullo 2014

D. Paulo Félix Lamas, Profesor do Departamento de Electrónica e Computación da Universidade de Santiago de Compostela, e **Dna. María José Carreira Nouche**, Profesora do Departamento de Electrónica e Computación da Universidade de Santiago de Compostela,

INFORMAN:

Que a presente memoria, titulada *Desenvolvemento dun entorno interactivo de baixo custo para a animación de personaxes utilizando dispositivos de captura de movemento sen marcas*, presentada por **D. Roberto Noya Noya** para superar os créditos correspondentes ao Traballo de Fin de Grao da titulación de Grao en Enxeñaría Informática, realizouse baixo nosa dirección no Departamento de Electrónica e Computación da Universidade de Santiago de Compostela.

E para que así conste aos efectos oportunos, expiden o presente informe en Santiago de Compostela, a 7 de Xullo de 2014:

O director, A codirectora, O alumno,

Paulo Félix Lamas María José Carreira Nouche Roberto Noya Noya

Agradecementos

A Xela, por estar aí todos estes anos, polo seu apoio continuo e, sobre todo, pola súa paciencia.

A Paulo e María José, por ser os mellores titores que un alumno poida desexar, por guiarme cando estiven perdido e por alentarme nos momentos de incerteza.

A Goose, polos seus consellos e por ser ese amigo incrible co que sempre se pode contar.

Á miña familia, por seguir estando aí a pesar das miñas ausencias continuadas, e en especial, a miña nai, que é a mellor.

Aos compañeiros e compañeiras da carreira, sen excepción, polas longas horas de aprendizaxe, terrario, *Lexín* e risas que me brindaron.

A *Tarlogic*, por terme dado a oportunidade de compaxinar o traballo con eles coa realización deste proxecto.

A todos eles, graciñas!!!

Índice xeral

Capítulo 1

Introdución

Nos inicios do cinema de debuxos animados, a partires de 1920, os encargados de darlle vida ás personaxes animadas eran debuxantes. Estes profesionais eran capaces de xerar unha animación debuxando, fotograma a fotograma, o movemento que unha personaxe debía realizar (Fig. 1.1). Esta técnica foi amplamente utilizada nos anos posteriores, dando lugar a múltiples producións, entre as que destacan as da factoría *Disney* [2].

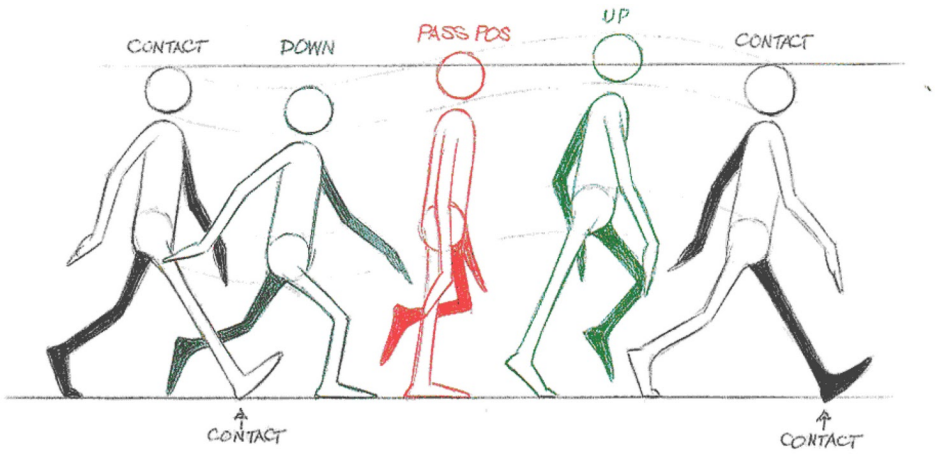

Figura 1.1: Fotogramas de animación debuxados por Richard Williams [1].

A finais da década dos setenta xorden as primeiras técnicas para xerar imaxes ou gráficos por ordenador. Posteriormente, en 1981, aparece no filme *Looker* (M. Crichton, 1981) a primeira personaxe xerada por ordenador, pero non é ata 1985 cando, na película *Young Sherlock Holmes* (B. Levinson, 1985) se mostra, por primeira vez no cine, unha personaxe animada xerada integramente por ordenador (Fig. 1.2). A partires dese momento, a maneira de entender o cine de animación e os efectos especiais no cine de imaxe real cambia en consonancia coa evolución das técnicas de xeración de imaxes e de animación por ordenador [3].

1.1. A animación por ordenador

No mundo da xeración de gráficos por ordenador, que abrangue dende a creación de longametraxes e curtametraxes ata a produción de videoxogos para diferentes plataformas, existen dúas aproximacións para levar a cabo a animación das personaxes:

Figura 1.2: Fotograma do filme *Young Sherlock Holmes* (B. Levinson, 1985), onde se mostra por primeira vez no cine unha personaxe xerada integramente por ordenador.

- A primeira xorde da utilizada polos primeiros animadores, polo que se pode denominar **tradicional**. Nesta técnica un animador ou equipo de animadores encárgase de darlle vida a unha personaxe de forma **manual**, pero coa axuda dun ordenador. Así, estes animadores contan con ferramentas software que lles permiten traballar sobre un **modelo poligonal** como o da figura 1.3, que consiste nunha malla de polígonos que se pode deformar. Desta maneira non é necesario debuxar todos os fotogramas que compoñen a animación senón que se pode acadar o mesmo resultado deformando o modelo ao longo do tempo. Ademais, estas ferramentas permiten definir puntos intermedios do

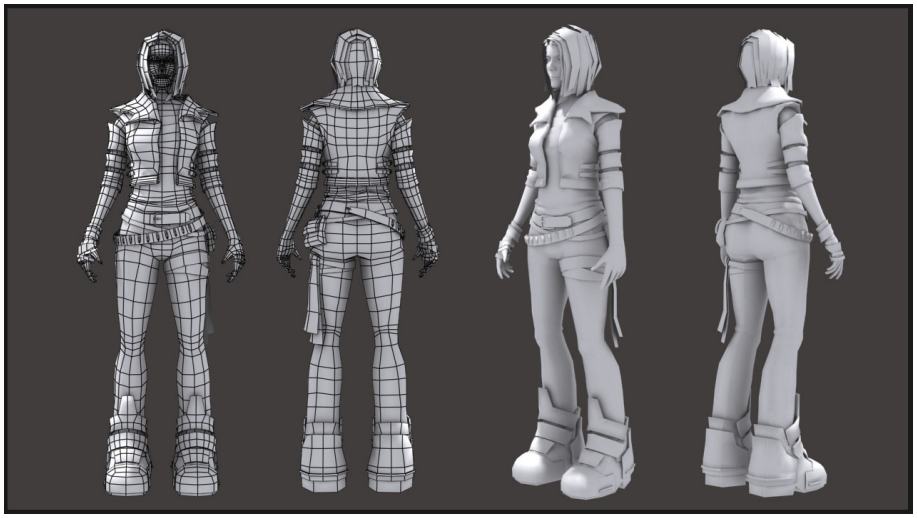

Figura 1.3: Modelo poligonal que se utiliza de base para realizar animacións por ordenador [4].

movemento, sendo o ordenador o que se encarga de xerar os fotogramas entre eses puntos. Esta primeira aproximación permite unha grande flexibilidade en canto ao resultado, podendo acadarse unha animación máis acorde coa realidade ou, polo contrario, un resultado máis no estilo dos debuxos animados.

- A segunda aproximaxión consiste na **captura de movemento** para a xeración das animacións de personaxes. Neste caso é necesario un sistema de captura de movemento, un actor ou grupo de actores para a realización das interpretacións e un equipo encargado de procesar estas interpretacións e convertelas en animacións, integrándoas coas personaxes dentro da produción. Na figura 1.4 pódese ver a complexidade do traxe de captura de movemento utilizado para a animación da personaxe *Gollum* na triloxía de *The Hobbit* (P. Jackson, 2012, 2013, 2014).

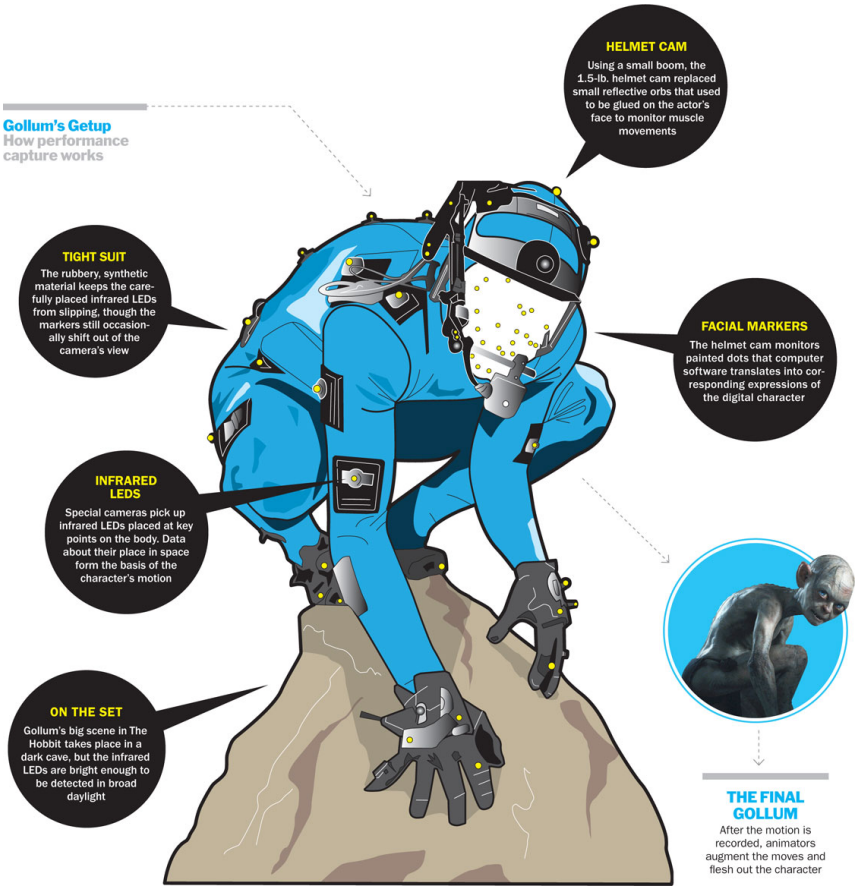

Figura 1.4: Traxe de captura de movemento utilizado para a animación da personaxe de *Gollum* na triloxía de *The Hobbit* (P. Jackson, 2012, 2013, 2014) (Ilustración de H. Jones para TIME, 2012 [5])

Moitas producións xeradas enteiramente por ordenador utilizan animadores profesionais seguindo a primeira aproximación, pero a contratación dun destes animadores non é algo ao que unha empresa pequena ou un traballador *freelance* teña fácil acceso en termos económicos. Ademais, en equipos pequenos o feito de animar un número grande de personaxes supón unha porcentaxe importante do tempo da produción, podendo condicionar en grande medida o resultado da mesma.

Se para unha produción se busca un estilo realista pódese optar por utilizar a segunda aproximación, pero o problema dos sistemas de captura de movemento profesionais é o seu elevado prezo, facendo complicada, ou incluso imposible, a súa adquisición para profesionais con recursos económicos limitados.

Nos últimos anos aparecen unha serie de dispositivos no ámbito das consolas de videoxogos que saen ao mercado como alternativa aos mandos tradicionais, propoñendo que o control do xogo o exerza o propio usuario cos movementos do seu corpo (Fig. 1.5). Estes dispositivos de baixo custo pódense considerar sistemas de captura de movemento, e é precisamente neles nos que se apoia a concepción do presente proxecto. Para contextualizalos afondaremos nas diferentes técnicas de captura de movemento que existen, para logo realizar unha pormenorizada análise do estado da técnica que permita entender cal é o lugar actual que estes dispositivos ocupan a día de hoxe.

Figura 1.5: Persoas utilizando o seu propio corpo para xogar co dispositivo de captura de movemento *Kinect* de *Xbox360* de *Microsoft* [6].

1.2. A captura de movemento

A captura de movemento é unha tecnoloxía relativamente nova que, coa proliferación na industria do cine dos gráficos xerados por ordenador e o salto cualitativo e cuantitativo da industria do videoxogo, gañou unha grande relevancia nos últimos anos.

Existen diferentes tipos de sistemas que se utilizan para obter datos de movemento, con importantes diferencias técnicas. Están os sistemas **inerciais**, que capturan o movemento utilizando acelerómetros e xiroscopios; os sistemas **mecánicos**, que utilizan un exoesqueleto para obter datos de movemento; os sistemas **magnéticos**, que calculan as posicións e orientacións en base ao fluxo magnético entre uns transmisores colocados nos puntos relevantes do corpo e os receptores correspondentes; e por último os sistemas **ópticos**, os máis comúns, que procesan a imaxe que captan os dispositivos de visualización para obter as posicións de determinados puntos de interese no tempo.

Explicaremos con máis detalle os sistemas ópticos posto que son os máis utilizados actualmente. A día de hoxe, os **sistemas ópticos** pódense dividir en dous grandes grupos:

- O primeiro e máis habitual denomínase sistema de captura de movemento **con marcas** (Fig. 1.6). As marcas son pequenos obxectos construídos cun material

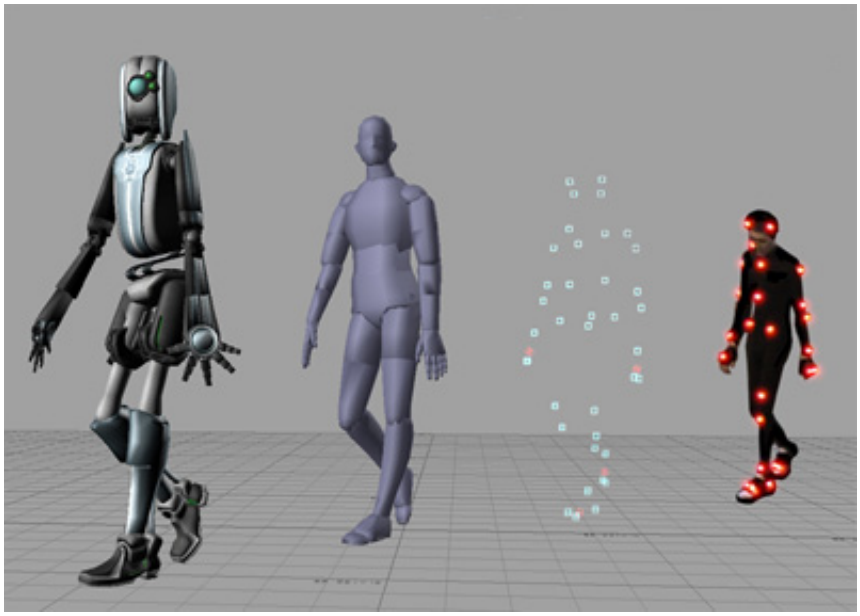

Figura 1.6: Exemplo de sistema óptico de captura de movemento con marcas [7].

que os dispositivos de captura son capaces de distinguir do resto da escena. Estas marcas colócanse nas articulacións do suxeito sobre o que se realiza a captura de movemento, de maneira que coas súas posicións no espazo ao longo do tempo é posible construír unha animación. É habitual que estas marcas se coloquen nun traxe elástico, que pode ser de corpo completo ou das partes do corpo que se desexan capturar, de maneira que o suxeito ao poñerse o traxe xa ten as marcas nos lugares correctos, o que fai moito máis fluída a preparación da captura.

- O segundo, máis moderno e cada vez máis en alza, denomínase sistema de captura de movemento **sen marcas** (Fig. 1.7). Estes sistemas baséanse en

Figura 1.7: Exemplo de sistema óptico de captura de movemento sen marcas (Organic Motion) [14].

algoritmos que analizan fluxos de vídeo e profundidade tratando de identificar formas humanas. Para utilizar estes sistemas, os suxeitos non necesitan levar marcas nin traxe especial, pero o lugar de captura debe ser nun contorno limitado en canto a iluminación, tamaño e fondo, restándolle un pouco de flexibilidade con respecto aos sistemas con marcas.

Como xa se comentou, nos últimos anos irrompen no mercado unha serie de dispositivos de captura de movemento de baixo custo no ámbito das consolas de videoxogos. Estes dispositivos permiten que o usuario interactúe co videoxogo empregando os movementos do seu propio corpo en lugar de utilizar un mando ou

calquera outro periférico. En pouco tempo xorden implementacións, configuracións, e pouco despois librarías, que permiten a conexión destes dispositivos a un ordenador, podendo recompilar, procesar e traballar con todo tipo de datos que estes son capaces de capturar.

Un dos aspectos máis interesantes destes dispositivos é que son capaces de capturar en vídeo a escena que teñen en fronte enriquecida cunha terceira dimensión; isto é, cada píxel do vídeo levará información da profundidade con respecto ao dispositivo do obxecto que é representado por ese píxel (Fig. 1.8). Algunhas destas

Figura 1.8: Información de vídeo e profundidade desde o dispositivo de captura de movemento sen marcas *Kinect* (*Microsoft*, 2010).

implementacións van máis aló, conseguindo que un destes dispositivos detecte o corpo humano, e acadando datos en tres dimensións de cada unha das partes do corpo ao longo do tempo. Conseguen así capturar o movemento dunha persoa sen a utilización de marcas e por un prezo moito máis reducido que os sistemas de captura de movemento do mercado.

1.3. Estado da técnica

Neste punto é necesario definir o espazo onde se encadra este proxecto dentro do contexto da captura de movemento, e o estado no que esta se atopa no momento actual.

En primeiro lugar, é necesario falar dos **sistemas ópticos con marcas**, porque aínda que son certamente máis caros seguen sendo os máis utilizados pola industria. A infraestrutura que os sistemas ópticos con marcas poñen en marcha é, no mellor dos casos, complexa, xa que son necesarias cámaras especiais, traxes con

marcas, software específico para cada un dos sistemas e o acceso a un contorno controlado, adaptado ás necesidades da animación a realizar. Toda esta infraestrutura leva consigo un custo que se reflicte no prezo final destes sistemas.

Neste ámbito atópanse grandes compañías como *Vicon* [8] ou *Qualisys* [9], que contan con sistemas que poden ir desde as decenas de miles de dólares ás centenas de miles. Baixando un nivel neste tipo de empresas, dentro sempre da captura de movemento con marcas, encádranse empresas como *Naturalpoint* [10], *XSens* [11], *Synertial* [12] ou *Xcitex* [13], con prezos que van desde miles a decenas de miles de dólares.

É habitual que estas compañías ofrezan os seus sistemas de maneira modular, dando ao cliente a flexibilidade de acadar a solución mellor adaptada ás súas necesidades e a escalabilidade para mellorar o seu sistema no futuro. É por esta razón que non se pode falar en termos de prezos concretos senón de rangos de prezos, e tamén por esta razón estes rangos son tan amplos.

En segundo lugar, como tecnoloxía emerxente temos os **sistemas ópticos sen marcas**, cuxo máximo expoñente na industria é a compañía *Organic Motion* [14]. Os resultados que acadan estes sistemas contan con bastante precisión, tendo entre os seus clientes grandes compañías como *Warner Bros* [15] dentro da industria do cine ou *Ubisoft* [16] na industria dos videoxogos. Pero estes sistemas non son asumibles para unha pequena produtora, xa que os prezos son moi similares aos dos sistemas con marcas, preto das decenas de miles de dólares, variando considerablemente o prezo segundo a complexidade do sistema adquirido.

Dentro deste segundo grupo tamén se encadra un grupo de empresas que, facendo uso de dispositivos de captura de movemento sen marcas de baixo custo, como pode ser o *Kinect* de *Microsoft* ou o *Xtion* de *Asus* (Fig. 1.9), ofrecen solucións software para a captura de movemento por un prezo moito máis asumible que as anteriores compañías.

Unha destas empresas é *Easy Mocap* [17], que ofrece un paquete software de captura de movemento cun só *Kinect* desde 700 dólares. Pero a empresa máis coñecida e con máis percorrido é *IpiSoft* [18], que ofrece a opción de traballar con ata dous *Kinect* ou *Xtion*, desde un prezo que ronda os 600 dólares.

Nos últimos anos houbo múltiples intentos de utilizar estes dispositivos de baixo custo cun software de xeración de gráficos por ordenador, na maioría dos casos *Blender* [22]. O proxecto máis consolidado é *Ni-Mate* [19], unha solución gratuíta

Figura 1.9: Dispositivos de captura de movemento sen marcas de baixo custo: *Asus Xtion* e *Microsoft Kinect*.

que permite a captura de movemento en tempo real para varias aplicacións, entre elas *Autodesk Maya* [21] e *Blender*. A captura realízase cun só *Kinect* ou *Xtion*. Outro proxecto a salientar é o traballo realizado por *G. Fahim* na universidade de *Gothenburg* (Suecia) [20], no que desenvolve un sistema co que é posible realizar captura de movemento utilizando un *Kinect* para obter unha animación en *Autodesk Maya* e *Motion Builder* [23] de maneira asíncrona.

1.4. Motivación do proxecto

O problema que aborda este proxecto é o de crear **animacións en tempo real** a partires da captura de movemento dunha persoa, tendo sempre en conta a premisa de que o sistema debe ser de **baixo custo** para posibilitar o seu acceso a pequenas empresas ou traballadores *freelance*.
Para obter unha animación o máis fiel posible, o sistema deberá tratar de minimizar na medida do posible as **oclusións**: ao realizar determinados movementos ocúltanse ao dispositivo certas partes do corpo da persoa. O feito de utilizar un só dispositivo de captura de movemento para a recollida de datos pode dar lugar a ditas oclusións, o que obriga ao sistema a inferir datos, en moitos casos de maneira incorrecta.

A maioría dos produtos comerciais mencionados traballan con tecnoloxías que minimizan ou eliminan por completo as oclusións no momento da captura, pero os prezos destas solucións afástanse moito do que se pode considerar baixo custo, polo que estarían descartados para resolver o problema proposto.

Como xa se comentou anteriormente existen alternativas moi interesantes e cun prezo máis acorde coa premisa do baixo custo, sen embargo dentro destas so-

lucións as que son gratuítas e de código aberto non permiten conectar máis dun dispositivo para minimizar as oclusións, e as que son comerciais e de código propietario non permiten a xeración da animación en tempo real.

É precisamente nese espazo onde se encadra este proxecto. Un sistema escalable en canto a número de dispositivos de captura para a recollida de datos permitirá ampliar o ángulo de visualización do sistema, evitando as oclusións e dotando de máis precisión aos datos recollidos.

1.5. Obxectivos do proxecto

O principal obxectivo deste proxecto é cubrir a necesidade dun método de baixo custo para a xeración de animacións por ordenador co deseño, desenvolvemento e proba dun sistema que permita:

- A adquisición e representación do movemento dun actor mediante unha configuración de un ou dous dispositivos de captura de movemento sen marcas de baixo custo.
- A transformación en tempo real desta representación na animación dunha personaxe no contorno dun software de xeración de gráficos por ordenador.
- O almacenamento da representación capturada para a súa posterior utilización.

O sistema deberá contar co mínimo acoplamento posible entre os diferentes módulos para poder adaptalo a calquera software de xeración de gráficos por ordenador.

1.6. Organización do documento

No presente documento detállanse as metodoloxías e procedementos empregados para o cumprimento dos obxectivos do proxecto.

No **Capítulo 2** abórdanse os aspectos relativos á xestión do proxecto: a planificación temporal, a metodoloxía de desenvolvemento, e a xestión tanto de riscos como da configuración.

No **Capítulo 3** preséntase o proceso de análise do software, comezando pola identificación de casos de uso para a partires destes realizar a extracción e especifi-

cación dos requisitos. A continuación, enúnciase o alcance do proxecto para rematar mostrando a organización do traballo nas diferentes iteracións realizadas.

No **Capítulo 4** realízase unha análise das opcións en canto ás tecnoloxías que poden ser utilizadas para cumprir os obxectivos do proxecto, para a continuación especificar cales formarán parte do entorno de traballo.

No **Capítulo 5** abórdase o deseño de software, comezando pola especificación da arquitectura do sistema e detallando, a partires de aí, o deseño proposto para cada unha das partes da mesma.

No **Capítulo 6** afóndase na parte experimental do proxecto, explicando en detalle os problemas técnicos que foron xurdindo durante o desenvolvemento e a maneira na que se resolveron.

No **Capítulo 7** móstrase o conxunto de probas deseñadas para ser realizadas por parte do cliente ao final de cada iteración. Tamén se presenta unha lista onde se mostra a validez de cada un dos requisitos e o seu grao de cumprimento.

No **Capítulo 8** expóñense tanto as conclusións extraidas do desenvolvemento do proxecto, como unha proposta de ampliacións futuras a partires do traballo realizado.

Capítulo 2

Xestión do proxecto

Neste capítulo abórdanse os aspectos relativos á xestión do proxecto: a planificación temporal, a metodoloxía de desenvolvemento, e a xestión tanto de riscos como da configuración.

2.1. Metodoloxía de desenvolvemento

Un dos primeiros pasos a dar para a realización dun proxecto consiste en decidir que metodoloxía se vai seguir para levar a cabo o traballo. Unha decisión pouco acertada ou errónea nesta cuestión pode supoñer atrasos, sobrecustos, ou incluso a non consecución dun proxecto. Polo tanto, é importante avaliar as condicións nas que se vai levar a cabo o proxecto, o nivel de incerteza, e a experiencia do equipo para escoller a metodoloxía máis adecuada.
Existe un importante grupo de metodoloxías denominadas **preditivas** que se basean nunha planificación inicial que se seguirá de maneira estrita durante todo o desenvolvemento, e nun catálogo de requirimentos que non varía. Por outra banda, existen as metodoloxías denominadas **áxiles**, que plantexan o traballo asumindo que vai haber cambios e variacións nos requisitos ao longo do desenvolvemento. Neste tipo de metodoloxía o contacto co cliente é continuo: asúmese que hai riscos que van ter lugar e permítese unha xestión máis eficiente dos mesmos. En resumo, unha metodoloxía áxil permite unha flexibilidade que unha preditiva non contempla.

O nivel de incerteza neste proxecto é certamente alto. Ademais, o equipo de desenvolvemento está formado por unha única persoa cun nivel de experiencia reducido en proxectos deste tamaño, e máis concretamente, na área de coñecemento na que se encadra este proxecto. Por esta razón, descártanse as metodoloxías preditivas en favor de metodoloxías áxiles.

2.1.1. Scrum - Metodoloxía áxil

Dentro das metodoloxías áxiles **Scrum** é unha das máis coñecidas, e sobre todo, unha das máis utilizadas. Este é un argumento de peso para considerala xa que, se o seu uso está en auxe e, sobre todo, se está substituíndo cada vez con máis forza ás metodoloxías preditivas, pódese asumir que os resultados serán, cando menos, satisfactorios.

Fundamentos de Scrum

- **Desenvolvemento iterativo:** *Scrum* traballa en iteracións coñecidas como *sprints* de curta duración, entre unha e catro semanas. Ao inicio de cada *sprint* propóñense unha serie de requisitos que se deben satisfacer, e faise unha estimación de canto tempo vai levar. O resultado dun *sprint* é un entregable funcional que formará parte do produto final.

- **Contacto continuo co cliente:** Ao rematar cada *sprint* convócase unha reunión co cliente, na que se proporciona un entregable e unha batería de probas que confirman que os requisitos implicados no *sprint* foron satisfeitos.

- **Desenvolvemento adaptativo:** Unha vez rematado o *sprint* e realizada a reunión co cliente, este debe proporcionar as súas impresións, críticas construtivas ou cambios que considere necesarios. Con este material o equipo reescribe o catálogo de requisitos para satisfacer as necesidades do cliente. Isto permite que o produto final se achegue moito máis ao que o cliente quere ou necesita.

- **Priorización dentro do catálogo de requisitos:** Os requisitos catalóganse cun nivel de importancia que ven dado polo peso que teñen dentro da consecución dos obxectivos do proxecto. Tamén se poden priorizar certos requisitos, aínda que en si mesmos non resulten de especial relevancia, se forman parte ou son necesarios para satisfacer un de maior importancia.

- **Mantemento dun ritmo de traballo:** Os prazos propostos para os *sprints* deben ser fixos: isto favorece a priorización das tarefas dentro do *sprint* e forza a tomar decisións no caso de que non se vaian cumprindo os obxectivos propostos.

Partes dun Sprint

- **Planificación ou *Sprint Planning*:** A primeira tarefa dun *sprint* é escoller os requisitos de maior importancia/relevancia, de entre os que aínda non se satisfixeron. Unha vez escollidos divídense nas tarefas necesarias para satisfacelos, e estímase canto tempo vai levar tanto cada tarefa como o *sprint* completo. Para completar o inicio do *sprint* é necesario deseñar as probas que se lle mostrarán ao cliente, demostrando que os requisitos involucrados son satisfeitos polo entregable.

- **Execución:** As tarefas definidas no *Sprint Planning* lévanse a cabo anotando en todo momento que tarefa está en progreso e que tarefas están completadas, e anotando o tempo real de duración de cada unha para poder comparar coas estimacións. Isto axuda a mellorar en estimacións para tarefas ou proxectos futuros.

- **Avaliación ou *Sprint Review*:** Organízase unha reunión co cliente na que se realizan as probas propostas. Como xa se comentou anteriormente, o cliente da a súa opinión sobre o resultado, que é coidadosamente anotada para realizar os cambios necesarios no catálogo de requisitos. Seguindo as opinións do cliente é posible que sexa necesario replanificar por cambios na prioridade de certos requisitos, e sempre para que o resultado final se adecúe o máis posible ao que o cliente necesita.

2.1.2. Aplicación da metodoloxía

Como *Scrum* está pensado para equipos de varias persoas, e este proxecto vai ser realizado por unha soa, hai conceptos desta metodoloxía que é complicado levar a cabo.

Este proxecto realizarase no entorno dun Traballo Fin de Grao, polo só se tomarán de *Scrum* aqueles conceptos que resulten beneficiosos para levar a cabo o proxecto en tempo e forma.

2.2. Planificación temporal

Para a elaboración da planificación do traballo tratouse de respectar, na medida do posible, a estrutura de desagregación do traballo presentada no anteproxecto [25]. Os cambios que se poden observar entre ambas xorden dun maior coñecemento do problema que é necesario resolver e das posibles maneiras de abordalo. No que respecta ás estimacións temporais realizadas, establécese unha xornada laboral de **20 horas semanais** para compaxinar o desenvolvemento do proxecto co traballo a media xornada que realiza o equipo de traballo.

2.2.1. EDT: Estrutura de Desagregación do Traballo

Para acadar unha planificación temporal coherente construirase en primeiro lugar a **EDT** (Fig. 2.1) do proxecto partindo do enunciado do alcance e, como xa se comentou, tendo presente a presentada no anteproxecto. Segundo o apartado 5.3 do PMBOK [24] o EDT outorga unha visión global desde o máis xeneral ao máis específico do alcance do proxecto e mostra dunha maneira xerárquica o traballo que é necesario levar a cabo para completar o mesmo.

2.2.2. Diagrama de *Gantt*

A desagregación do traballo non debe ser vista como unha organización cronolóxica, senón máis ben como un punto de partida para a segunda parte desta planificación temporal, o **diagrama de *Gantt*** (Fig. 2.2). As diferentes tarefas nas que se desagrega o traballo presentes na *EDT* trasladaranse a este diagrama de *Gantt* mostrando, neste caso, a organización temporal de todas elas dentro do tempo de realización do proxecto.

2.2.3. Fases principais do proxecto

A continuación móstranse as fases do *EDT* de mais alto nivel, indicando en cada caso cal é a natureza as tarefas que abarca e o tempo, tomando como referencia o *diagrama de Gantt*, que se considera necesario para a súa realización:

Fase de *Xestión do proxecto*

Duración: 54.25 horas

Descrición: Esta fase abarca todas aquelas tarefas que teñen incidencia no proxecto como tal. Por esta razón, é unha fase que ten tarefas ao longo de todo o tempo do proxecto. Nesta fase sitúanse desde o enunciado do alcance do proxecto ata as tarefas de documentación, pasando pola creación do *EDT* e o diagrama de *Gantt*.

Fase de *Preparación*

Duración: 47 horas

Descrición: A fase de preparación xorde polo descoñecemento por parte do equipo de desenvolvemento das tecnoloxías e ferramentas coas que se debe levar a cabo o proxecto e ten como obxectivo que o equipo aborde o desenvolvemento coa maior decisión e seguridade. Nesta fase tamén se estudarán diferentes metodoloxías de desenvolvemento para escoller a mais adecuada para o presente proxecto.

Fase de *Iniciación*

Duración: 46 horas

Descrición: Na fase de iniciación proponse unha descrición mais detallada do sistema, realizando un proceso de captura de requisitos. Comezarase pola extracción de casos de uso a partires das entrevistas co cliente, que valerán como punto de partida para extraer e especificar o requisitos que definen ao sistema. A continuación realizarase unha planificación temporal, tendo a análise preliminar e a metodoloxía de desenvolvemento escollida. Por último, abordarase o deseño preliminar do sistema, comezando por definir a arquitectura do mesmo, para rematar especificando o deseño de cada unha das partes en dita arquitectura.

Fase de *Desenvolvemento*

Duración: 247.5 horas

Descrición: Neste fase levarase a cabo a codificación dos diferentes elementos que conforman o sistema. Este proceso estará dividido en iteracións de maneira que en cada unha delas se leve a cabo, non so a implementación do produto, senón tamén unha etapa de probas, nunha reunión co cliente, unha vez rematada dita iteración. A partires dos datos recollidos nesta reunión realizarase unha revisión do alcance, para comprobar que o produto segue cumprindo o que o cliente quere. Para abordar a nova iteración proponse unha etapa de análise e outra de deseño centradas na parte do sistema que se vai desenvolver, e tendo en conta se existen novos requisitos ou restriccións de deseño que é necesario ter en conta. Desta maneira os requisitos especificados na fase de iniciación evolucionan para adaptarse cada vez mais ás necesidades e desexos do cliente.

Fase de *Conclusión*

Duración: 4 horas

Descrición: Por último, na fase de conclusión realizase a integración de todos os elementos dos sistema que non foran integrados na fase de desenvolvemento, e realízanse as probas deseñadas para cada un dos sprints co produto final. Se o produto pasa as probas dase por rematado o proxecto e pódese proceder ao seu peche.

2.3. Xestión da configuración

A xestión da configuración nun proxecto como o que nos ocupa consiste nunha serie de medidas e procesos cuxo único propósito é manter a integridade dos elementos de traballo, entendendo por elemento de traballo tanto os ficheiros de código fonte e de configuración dentro do *IDE* ou entorno de desenvolvemento, como todos aqueles arquivos que compoñen a documentación do proxecto.

Para establecer un plan integral de xestión da configuración parece máis adecuado tratar por separado os dous tipos de elementos de traballo comentados, xa que as necesidades, no caso de perda de integridade, son diferentes en cada caso.

2.3.1. Control do código fonte

Existen ferramentas moi potentes para o control de cambios e control de versións de código fonte. A escollida para a xestión do código fonte neste proxecto é **Git**, na parte de control do repositorio local. Para que aumentar a seguridade e que esta non dependa totalmente do estado do ordenador de traballo, *Git* traballará contra un repositorio remoto aloxado en **BitBucket**.

A dinámica de traballo con *Git* e *BitBucket* será a seguinte: Cada vez que se remate unha tarefa das que compoñen o *sprint* en curso ou cada vez que se realice un cambio importante, levarase a cabo un *Commit* ao repositorio local. Un *Commit* garda unha instantánea do estado do proxecto no repositorio local. Por outra banda, unha vez remate o día de traballo ou cando se remata unha parte importante do sistema realizarase un *Push* que subirá todos os cambios e avances (*Commits*) realizados desde o anterior *Push*, ao repositorio remoto. Desta maneira se

existe un problema co ordenador de traballo sempre haberá unha copia no repositorio remoto.
Cabe salientar que *Git* non permite facer un *Commit* sen introducir un comentario cos cambios ou avances realizados. É unha boa práctica comentar detalladamente en cada *Commit* todo o que se acadou desde o anterior, xa que servirá como folla de ruta da fase de implementación con tempos de duración das partes, fitos, etc. Esta información pode ser moi valiosa para a toma de decisións en proxectos futuros.

2.3.2. Control da documentación

Aínda que *Git* permite xestionar calquera tipo de ficheiro aparte dos de código fonte, cando se traballa con ficheiros que non son en texto claro, é máis cómodo que o control de cambios sexa xestionado automaticamente cada vez que o ficheiro en cuestión sufra unha modificación. Tendo isto en conta, unha ferramenta como **Dropbox** parece máis adecuada que *Git* para o control de cambios nos ficheiros de documentación do proxecto.
Dropbox proporciona espazo na nube para o almacenamento de ficheiros. Ao instalar a aplicación de escritorio de *Dropbox* créase unha carpeta local no ordenador que será sincronizada periodicamente coa carpeta remota aloxada nos servidores da compañía. Desta maneira os cambios en calquera ficheiro, sexa cal sexa a súa natureza ou extensión queda almacenado como versión anterior, podendo ser recuperado en calquera momento en caso de problemas de integridade ou cambios non válidos.
Para diminuír as posibilidades de perder traballo no caso dun problema de integridade, ademais da carpeta onde se almacena a documentación do proxecto, tamén se gardará o repositorio local de *Git* na carpeta de *Dropbox*. Así, se existe un problema con este repositorio, e tamén co repositorio remoto, sempre quedaría unha copia do código cos seus cambios en *Dropbox*.

2.4. Xestión de riscos

A xestión de riscos nun proxecto cun nivel de incerteza e experimentación tan grande como o que nos ocupa é esencial. O feito de identificar posibles eventos, circunstancias ou situacións que escapan ao control do persoal que leva a cabo o proxecto, e ter definidas unha serie de medidas que permitan prever, minimizar ou incluso eliminar as fontes de risco, pode supoñer a diferencia entre rematar o proxecto en tempo e forma, sen sobrecustos, ou, polo contrario, non cumprir os

prazos acordados ou que sexa necesaria unha revisión do custo final, reducindo a marxe de beneficios. Existen catro pasos a seguir na xestión de riscos:

- Identificación
- Análise e catalogación
- Planificación
- Seguimento

Unha vez identificados os posibles riscos que poden afectar a un proxecto é necesario analizalos un por un para poder catalogalos en base á súa relevancia no transcurso do traballo. Para avaliar esta relevancia utilizaranse tres medidores:

A **Probabilidade** de que un risco ocorra:

Baixa: Menos do 25 % de probabilidade de que o risco ocorra.

Media: Entre o 25 % e o 75 % de probabilidade de que o risco ocorra.

Alta: Máis do 75 % de probabilidade de que o risco ocorra.

O **Impacto** en tempo/esforzo/cartos que supón se un determinado risco ocorre:

Baixo: Menos do 10 % de repercusión en prazo/esforzo/custo.

Medio: Entre o 10 % e o 20 % de repercusión en prazo/esforzo/custo.

Alto: Máis do 20 % de repercusión en prazo/esforzo/custo.

E, por último, o nivel de **Exposición** que combina os valores de Probabilidade e Impacto, dando unha medida moi adecuada para saber a que riscos hai que prestarlles máis atención durante o proxecto:

		Probabilidade		
		Alta	Media	Baixa
Impacto	Alto	Alto	Alto	Medio
	Medio	Alto	Medio	Baixo
	Baixo	Medio	Baixo	Baixo

Táboa 2.1: Nivel de exposición dun risco en base a Probabilidade e Impacto

Como a xestión de todos os riscos faría moito máis complexo o seguimento do proxecto, restándolle tempo á propia elaboración do mesmo, so se planificarán e se fará un seguimento daqueles riscos cun nivel de Exposición Alto. Este nivel de Exposición indica que un risco é bastante probable que apareza e, ademais, no caso de aparecer, tería un impacto considerable no desenvolvemento do proxecto. Existen tres tipos de resposta posibles no caso de que un risco teña un nivel de exposición alto:

***Minimizar o risco*:** Consiste en reducir o impacto do risco

***Evitar o risco*:** Consiste en reducir a probabilidade de que un risco apareza

***Elaborar un plan de continxencia*:** Actuación no caso de que o risco apareza, para que afecte o menos posible

A continuación móstranse os riscos considerados, organizados en categorías segundo a parte do proxecto á que afectan:

2.4.1. Riscos na xestión do proxecto

RSC.01 - Análise de requisitos errónea

Descrición: A análise de requisitos non se axusta coas necesidades do cliente

Probabilidade: Media

Impacto: Alto

Nivel de Exposición: Alto

Resposta ao risco: A elección de *Scrum* como metodoloxía de traballo minimiza o impacto deste risco, xa que está contemplado que existan erros na captura inicial de requisitos e que estes se vaian refinando ao longo do tempo do proxecto cos contactos continuos co cliente.

RSC.02 - Incumprimento dos prazos de entrega

Descrición: Debido á falta de experiencia nas tarefas que é necesario levar a cabo é posible que se produzan atrasos nas entregas.

Probabilidade: Media

Impacto: Alto

Nivel de Exposición: Alto

Resposta ao risco: A elección de *Scrum* reduce a probabilidade de que ocorra este risco xa que o feito de ter reunións co cliente cada pouco tempo evita grandes atrasos.

2.4.2. Riscos de dispoñibilidade

RSC.03 - Licenza de Autodesk Maya

Descrición: Para a realización do proxecto é necesaria unha licenza de Autodesk Maya, un software comercial. O feito de non acadar unha licenza non permitiría cumprir con parte do proxecto.

Probabilidade: Alta

Impacto: Alto

Nivel de Exposición: Alto

Resposta ao risco: Minimizamos o impacto do risco solicítase a *Autodesk* unha licenza educacional gratuíta para levar a cabo o proxecto

RSC.04 - Fallo nun dispositivo de captura

Descrición: Un dos dispositivos de captura falla durante o desenvolvemento do proxecto cos conseguintes atrasos.

Probabilidade: Baixa

Impacto: Alto

Nivel de Exposición: Media

Resposta ao risco: Como plan de continxencia solicítase ao cliente a posibilidade de contar cun reposto no caso de que un dos dispositivos falle durante o desenvolvemento.

2.4.3. Riscos no desenvolvemento

RSC.05 - Viabilidade técnica do proxecto

Descrición: A solución proposta non é posible, ou é mais complexa tecnicamente do que se esperaba.

Probabilidade: Media

Impacto: Alta

Nivel de Exposición: Alto

Resposta ao risco: Para minimizar o impacto desenvolverase dun prototipo vertical ou proba de concepto co que se probarán as funcionalidades mais críticas para a consecución do proxecto.

RSC.06 - Fallo no ordenador de traballo

Descrición: Un fallo no ordenador de traballo produce atrasos e a perda do código do desenvolvemento.

Probabilidade: Baixa

Impacto: Alto

Nivel de Exposición: Medio

Resposta ao risco: Para reducir o impacto establécese unha xestión da configuración robusta. O uso de *Git* como repositorio e o feito de que os ficheiros de traballo estean aloxados en *Dropbox* asegura que en caso de fallo do ordenador de traballo, sexa posible recuperar este código a outro ordenador e continuar traballando.

2.4.4. Riscos de persoal

RSC.07 - Uso de tecnoloxías descoñecidas

Descrición: O equipo de desenvolvemento non ten experiencia nas ferramentas e tecnoloxías propostas para levar a cabo o proxecto.

Probabilidade: Alta

Impacto: Medio

Nivel de Exposición: Alto

Resposta ao risco: No plan de traballo do proxecto engádese unha fase de Preparación cuxo único obxectivo será que o equipo chegue coas competencias necesarias ao desenvolvemento do proxecto.

2.5. Análise de custos

Nesta sección detallaranse que recursos tanto humanos como de material son necesarios para levar a cabo o proxecto e cal é a previsión de custo para o cliente.

2.5.1. Recursos humanos

Para a realización do seguinte proxecto cóntase cun equipo de desenvolvemento dun só membro, que realizará tarefas de análise e desenvolvemento de software. Segundo *Infojobs* [26], o salario bruto medio na provincia de A Coruña para un analista programador é de aproximadamente *20.500 €*. Tendo en conta que o custo para a empresa é aproximadamente de 1,6 veces o soldo do empregado [27] [28], e que ó ano hai 1.800 horas laborais nunha xornada normal de 8 horas diarias, pódese concluír que o soldo medio por hora dun analista programador nesta provincia é de **18,22 €/hora**.

2.5.2. Recursos materiais

Para a consecución do proxecto é necesario un ordenador de traballo e dous dispositivos de captura de movemento *Kinect*. O prezo dun PC de sobremesa é de aproximadamente **900 €** e cada *Kinect* de aproximadamente **160 €**.

2.5.3. Recursos software

Para a realización deste proxecto utilizáronse as seguintes ferramentas software: *Visual Studio Express 2012*, para o desenvolvemento de software; *Texmaker*, para a realización da memoria; *Microsoft Visio*, para realización dos diagramas, e as librarías *SDK Microsoft*, *XNA* e *Math.net*. Todos estes recursos son gratuítos, excepto *Visio*, pero é posible utilizar unha licencia gratuíta para estudantes da *USC*.

2.5.4. Presuposto

A continuación detállase os gastos do proxecto:

Concepto	**Cantidade**	**Custo unitario**	**Custo Total**
Analista Programador	398,75 horas	18,22 €/hora	7265,22 €
PC sobremesa	900 €	1	900 €
Microsoft Kinect	160 €	2	320 €
		Total	**8485.22 €**

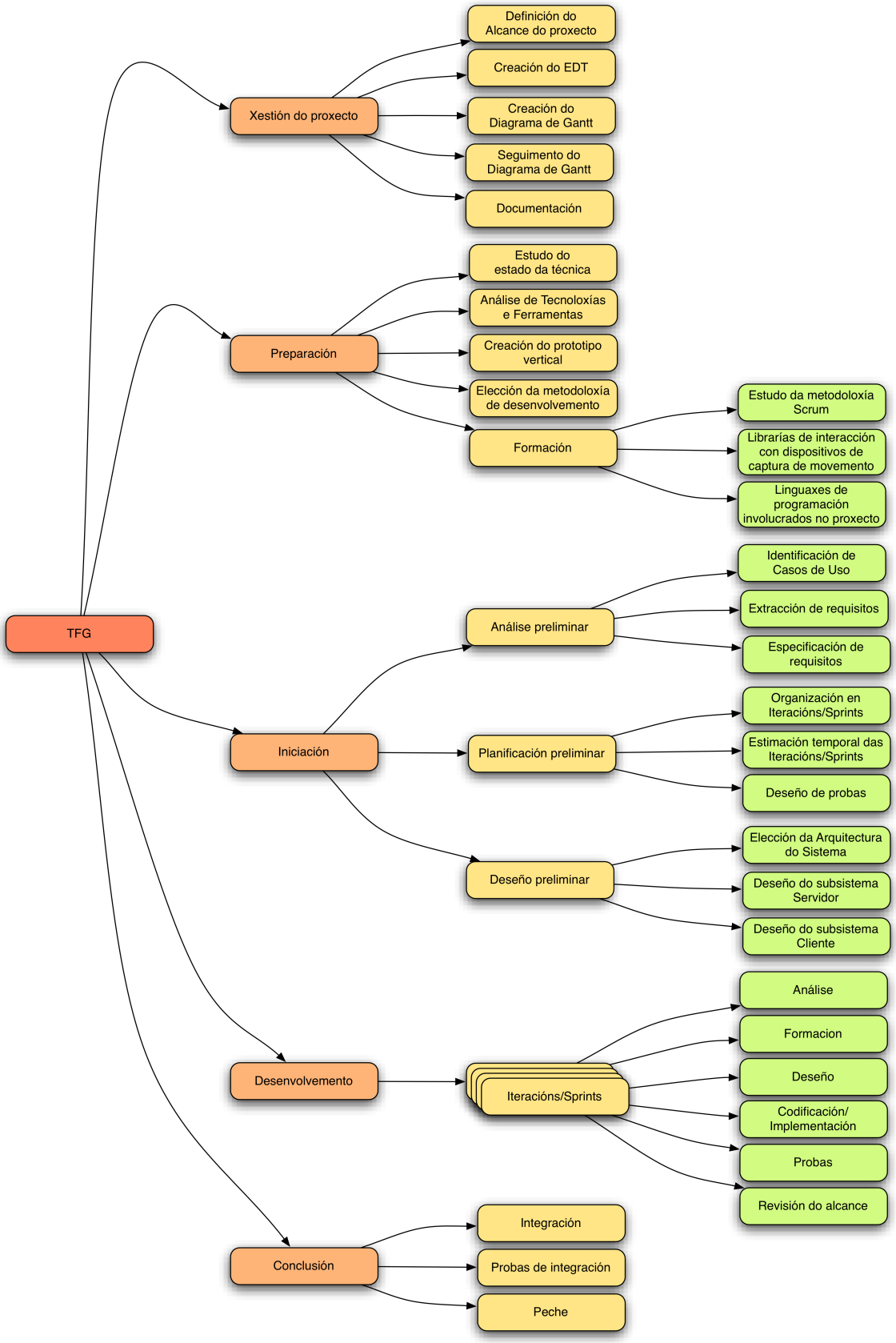

Figura 2.1: Estrutura de Desagregación do Traballo

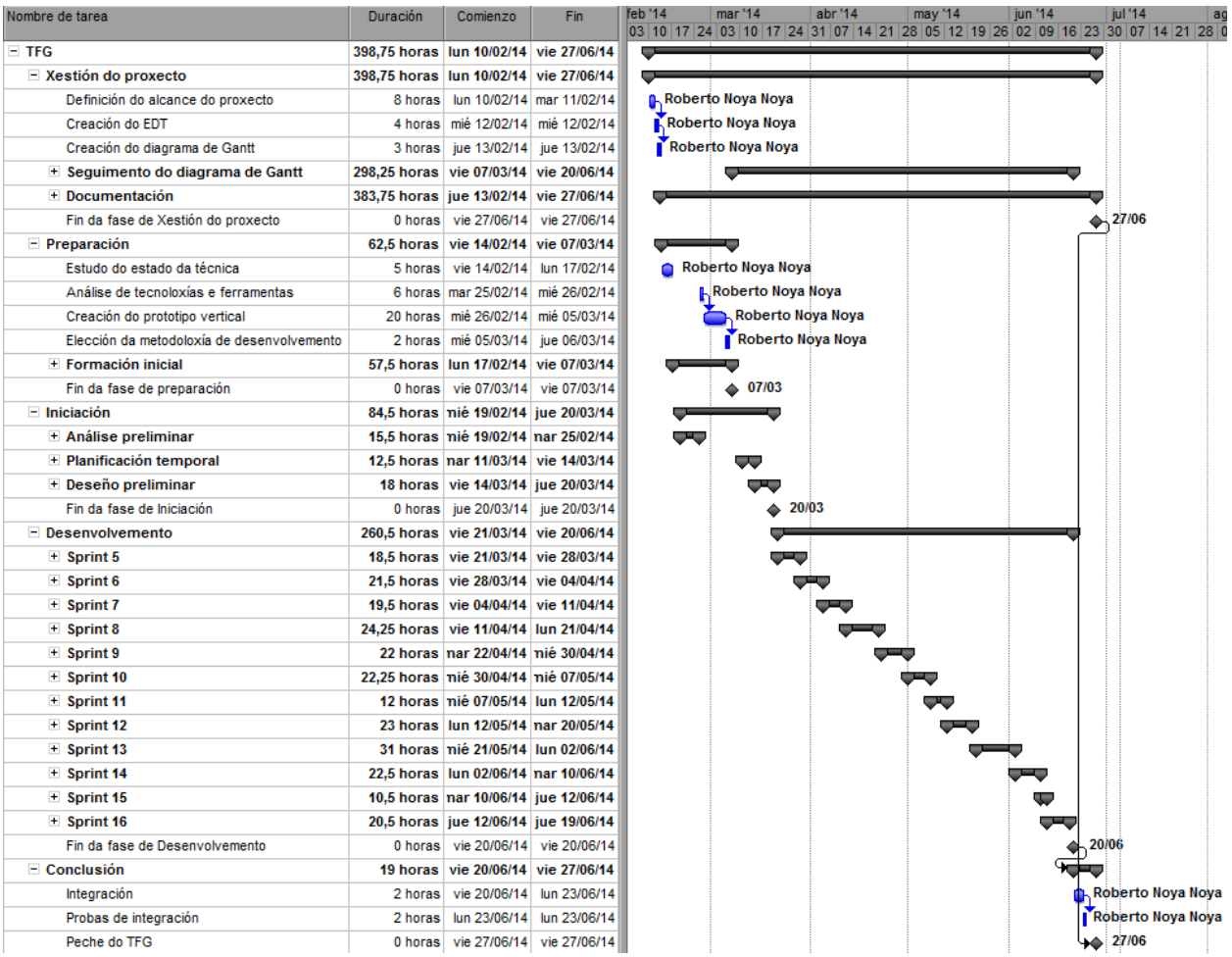

Figura 2.2: Diagrama de Gantt

Capítulo 3

Análise de requisitos

En toda metodoloxía de desenvolvemento, o primeiro paso para construír un novo produto de software é precisar que **requisitos** deberá cumprir este para satisfacer as necesidades do cliente. Comezamos por definir, coa axuda do cliente, as situacións nas que o software a desenvolver vai ser utilizado e de que maneira: os **casos de uso** do software. A partires destes casos de uso, extraemos un conxunto de requisitos funcionais e non funcionais que definen con maior precisión o que o cliente necesita do produto software.

3.1. Casos de uso

A extracción de casos de uso é unha técnica moi utilizada para a captura de información directamente do cliente. Os casos de uso describen dunha maneira informal e pouco técnica a interacción entre os usuarios do sistema, que no ámbito dos casos de uso se denominan **Actores**, e o sistema a desenvolver. O conxunto de todos os casos de uso debe ilustrar a alto nivel a maneira na que o cliente desexa que o sistema traballe, sen abordar cuestións de implementación ou comportamento.

Os Actores son representacións xenéricas dos tipos de usuario que van utilizar o sistema. A relación dun determinado Actor cos casos de uso define a maneira concreta na que ese tipo de usuario interactuará co sistema. Unha boa definición de Actores axuda a definir permisos e restricións de uso do sistema para segundo que usuarios.

3.1.1. Consideracións iniciais

Estas consideracións son extraidas directamente das entrevistas co cliente e axudan a contextualizar o traballo a realizar e certos conceptos específicos que se van manexar nos casos de uso.

O cliente precisa un sistema que cunha infraestrutura de captura de movemento de baixo custo e un software de xeración de gráficos por ordenador permita a gravación dos movementos dun intérprete, mediante o que a partires de agora denominaremos unha **sesión de interpretación**. Esta sesión de interpretación será a base para obter a animación dunha personaxe no contorno do software de xeración de gráficos por ordenador.

Dentro deste software é necesario crear unha **entidade** que almacene os datos

de animación. O máis común é que esta entidade sexa o que se coñece como *skeleton* ou esqueleto (Fig. 3.1), que está composto por unha xerarquía de articulacións ou *joints* en *Autodesk Maya*, e ósos ou *bones* en *Blender*. A partires deste momento denominaremos ás partes que compoñen o esqueleto **ósos**: cada un deles terá unha posición ou rotación en cada fotograma da animación. O feito de que o esqueleto sexa xerárquico implica que cando un óso cambia de posición ou rotación, todos os que están debaixo deste na xerarquía móvense da mesma maneira.

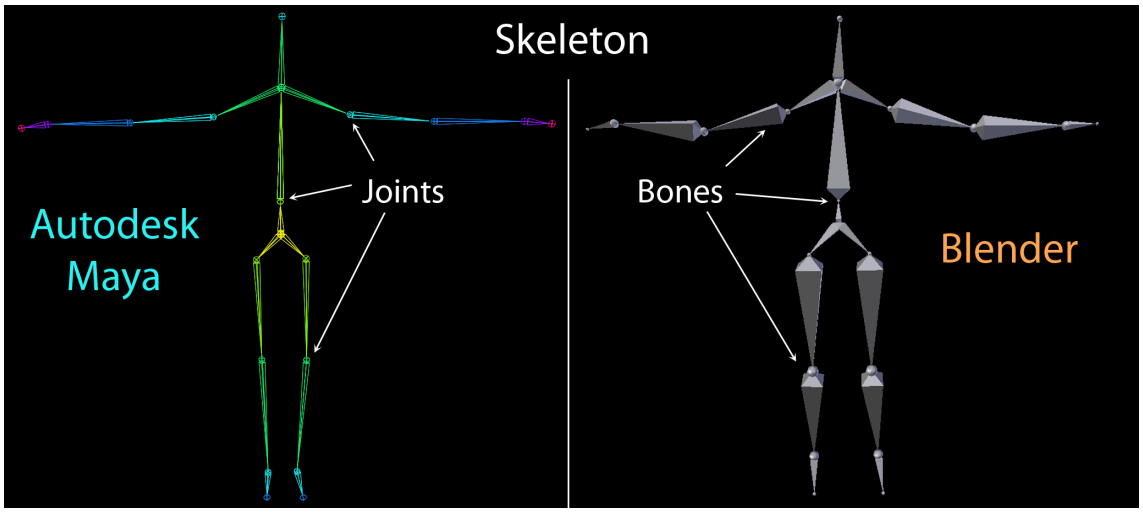

Figura 3.1: Representación das entidades que almacenan os datos de animación: esqueletos en *Autodesk Maya* e *Blender*.

Unha **animación** consiste en aplicar movemento ao longo do tempo aos diferentes ósos que forman o esqueleto. Esta animación pode estar baseada nas **posicións** absolutas no espazo que os ósos teñen no tempo, ou nas **rotacións** que estes teñen con respecto ao esqueleto. A animación baseada en posicións traballa con valores absolutos da posición no espazo de cada óso, polo que se torna máis complexo traballar con esqueletos de proporcións diferentes ás do intérprete. Por contra, a animación baseada en rotacións permite a flexibilidade que se bota en falta traballando con posicións. Neste caso, aínda que as proporcións do esqueleto sexan completamente diferentes ás do intérprete, estas vanse manter ao longo do tempo a pesar das rotacións que se lle poidan aplicar aos diferentes ósos.

Para deseñar o sistema é necesario ter en conta o problema das **oclusións** que se producen cando o intérprete, ao realizar determinados movementos, oculta ao dispositivo certas partes do seu corpo. Para minimizar este problema é posible utilizar alomenos dous dispositivos, situados de maneira que se amplíe o ángulo de

visualización do sistema. Desta forma a liberdade de movementos do intérprete é maior, permitindo máis flexibilidade para a creación de animacións.

Por último, é necesario considerar a posibilidade, na medida do posible, de que o sistema permita traballar en **tempo real**: isto é, mentres está tendo lugar a gravación da sesión de interpretación estase a crear a animación no software de xeración de gráficos por ordenador. Esta característica permite unha maior fluidez á hora de xerar animacións, xa que se pode revisar o resultado final, e en caso de que non sexa o desexado, repetir a sesión ata obter a animación adecuada.

3.1.2. Descrición de Actores

O usuario tipo que interactúa co sistema queda representado polo seguinte Actor:

Actor - ACT.01

Nome: Usuario

Descrición: Actor que representa a unha persoa que fai uso da aplicación

3.1.3. Descrición de Casos de Uso

As entrevistas co cliente mostran unha clara división do sistema en dúas partes ben diferenciadas:

- A gravación da sesión de interpretación, que unha vez instalado o sistema se realizara nunha estancia con suficiente espazo para aloxar ao dispositivo ou dispositivos de captura. A partires de agora denominaremos a esta parte **Estudio de Gravación**.
- A xeración da animación, que transcorre no ámbito dun software de xeración de gráficos por ordenador e que a partires de agora denominaremos **Estudio de Animación**.

Móstrase a continuación a definición e descrición dos casos de uso que se identificaron nas diferentes entrevistas co cliente, organizados segundo a parte na que se encadran. Axúntase tamén, na figura 3.2, o diagrama de casos de uso para unha mellor comprensión da estrutura.

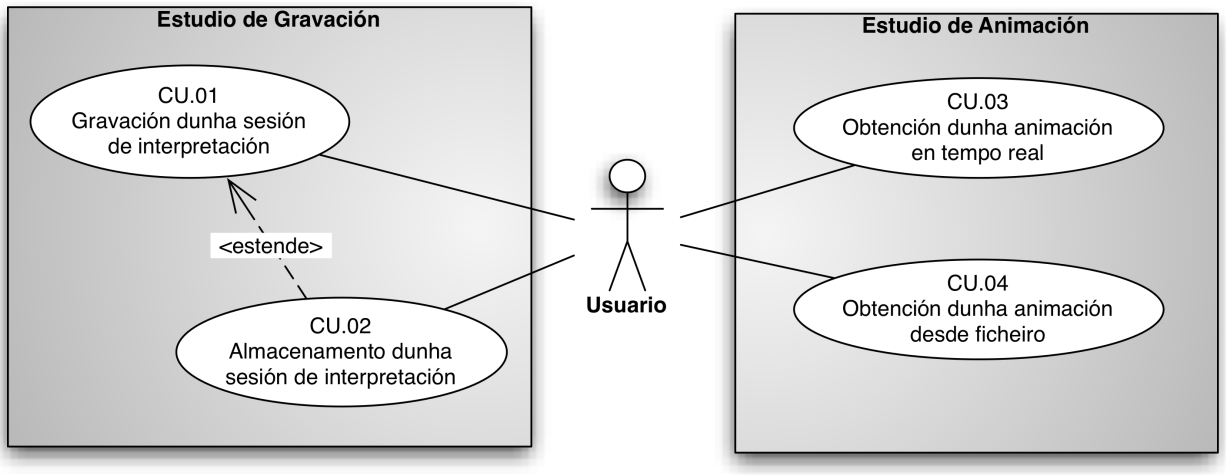

Figura 3.2: Diagrama de casos de uso

Estudio de Gravación

Caso de uso - CU.01

Nome: Gravación dunha sesión de interpretación

Propósito: Un usuario debe poder gravar unha sesión de interpretación realizada por un intérprete, que pode ser o propio usuario, utilizando un ou máis dispositivos de captura de movemento sen marcas.

Actores:

- Usuario

Relación con outros casos de uso:

- É estendido polo CU.02.

Precondicións: Non constan.

Poscondicións:

- A sesión de interpretación foi gravada.

Escenario principal:

O caso de uso comeza cando se desexa gravar unha sesión de interpretación, que será a base para xerar a animación dunha personaxe. O sistema pode ser utilizado por unha soa persoa que actúa como usuario e como intérprete.

O usuario conecta un dispositivo de captura de movemento de baixo custo ao

equipo e inicia o sistema, que recoñece que hai un dispositivo conectado. O intérprete sitúase fronte ao dispositivo de captura e o usuario inicia a gravación da sesión de interpretación interactuando co sistema por comando ou voz. O intérprete realiza a sesión de interpretación e cando remata, o usuario detén a gravación interactuando de novo co sistema por comando ou voz.

Escenario alternativo:

O usuario conecta dous dispositivos de captura de movemento de baixo custo ao equipo e inicia o sistema, que recoñece que hai dous dispositivos conectados. O sistema só permite a gravación dunha sesión de interpretación cando os dous dispositivos traballen como se fosen un só, polo que lle informa ao usuario de que é necesario calibrar. O intérprete sitúase de maneira que todo o seu corpo é captado polos dous dispositivos a un tempo e o usuario inicia a calibración interactuando co sistema por comando ou voz. Durante a etapa de calibración o intérprete realiza movementos tratando de cubrir o maior volume posible, procurando que o seu corpo enteiro siga sendo captado polos dous dispositivos ao mesmo tempo. A partires deste punto, o resto deste escenario transcorre de igual maneira que o escenario principal.

Subsistemas:

- Estudio de Gravación

Caso de uso - CU.02

Nome: Almacenamento dunha sesión de interpretación

Propósito: Un usuario debe poder almacenar nun ficheiro unha sesión de interpretación que ven de ser gravada.

Actores:

- Usuario

Relación con outros casos de uso:

- Estende ao CU.01.

Precondicións:

- A gravación da sesión de interpretación que se quere almacenar xa rematou.

Poscondicións:

- Os datos da sesión de interpretación quedan almacenados nun ficheiro cun formato acordado.

Escenario principal:

O caso de uso comeza cando un usuario, unha vez rematada a gravación dunha sesión de interpretación, necesita almacenar nun ficheiro os datos recollidos. Cando a gravación dunha sesión de interpretación remata o usuario almacena os datos de captura nun ficheiro interactuando co sistema por comando.

Subsistemas:

- Estudio de Gravación

Estudio de Animación

Caso de uso - CU.03

Nome: Obtención dunha animación en tempo real

Propósito: Un usuario debe poder obter unha animación en tempo real a partires dos datos de captura obtidos na gravación dunha sesión de interpretación no contorno dun software de xeración de gráficos por ordenador.

Actores:

- Usuario

Relación con outros casos de uso: Non consta.

Precondicións:

- A gravación dunha sesión de interpretación está en curso.
- O software de xeración de gráficos por ordenador está en execución no equipo.

Poscondicións:

- Obtense unha animación dentro do contorno do software de xeración de gráficos por ordenador a partires dos datos de captura recibidos.

Escenario principal:

O caso de uso comeza cando é necesario obter unha animación en tempo real no contorno dun software de xeración de gráficos por ordenador, a partires dos datos adquiridos na gravación dunha sesión de interpretación. Neste caso de uso o usuario interactúa co sistema a través de comandos dentro do propio software de xeración de gráficos por ordenador.

O usuario crea un esqueleto, entidade que o software de xeración de gráficos por ordenador utilizará para almacenar a animación. A continuación establece, se non son válidos os valores por defecto, o fotograma no que se vai iniciar a animación e especifica se a personaxe animada se debe trasladar no espazo ou debe permanecer no mesmo lugar durante a animación. Por último, o usuario inicia a obtención da animación. O sistema mostra a animación, representada polos movementos do esqueleto, ao mesmo tempo que se está a producir a gravación da sesión de interpretación.

Subsistemas:

- Estudio de Animación

Caso de uso - CU.04

Nome: Obtención dunha animación desde ficheiro

Propósito: Un usuario debe poder obter unha animación cargando un ficheiro que conteña os datos de captura gravados na sesión de interpretación.

Actores:

- Usuario

Relación con outros casos de uso: Non consta.

Precondicións:

- O ficheiro contén datos de captura.
- O software de xeración de gráficos por ordenador está en execución no equipo.

Poscondicións:

- Obtense unha animación dentro do contorno do software de xeración de gráficos por ordenador a partires dos datos de captura adquiridos.

Escenario principal:

O caso de uso comeza cando é necesario obter unha animación no contorno dun software de xeración de gráficos por ordenador, a partires dos datos adquiridos na gravación dunha sesión de interpretación e almacenados nun ficheiro. Neste caso de uso o usuario interactúa co sistema a través de comandos dentro do propio software de xeración de gráficos por ordenador.

O usuario crea un esqueleto, entidade que o software de xeración de gráficos por ordenador utilizará para almacenar a animación. A continuación establece, se non son válidos os valores por defecto, o fotograma no que se vai iniciar a animación e especifica se a personaxe animada se debe trasladar no espazo ou debe permanecer no mesmo lugar durante a animación. Por último, o usuario carga o ficheiro con datos de captura da gravación para obter a animación. O sistema mostra a animación, representada polos movementos do esqueleto ao longo do tempo.

Subsistemas:

- Estudio de Animación

3.2. Restricións de deseño

Neste punto especifícanse cales son as restricións que debe ter en conta o equipo de desenvolvemento á hora de abordar o proxecto. Estas restricións veñen impostas ben polo cliente, ben polas características do proxecto, que poden delimitar a maneira de levar a cabo o desenvolvemento.

Restrición de deseño - RDe.01

Título: Software Comercial: *Autodesk Maya*

Descrición: Desenvolvemento para o entorno do software comercial de xeración de gráficos por ordenador, *Autodesk Maya*.

Relevancia: Esencial

Restrición de deseño - RDe.02

Título: Software de código aberto: *Blender*

Descrición: Desenvolvemento para o entorno do software de código aberto de xeración de gráficos por ordenador, *Blender*.

Relevancia: Esencial

Restrición de deseño - RDe.03

Título: Método de almacenamento: Ficheiro en texto plano

Descrición: O método de almacenamento será un ficheiro en texto plano.

Relevancia: Alta

Restrición de deseño - RDe.04

Título: Implementación dunha cola para o almacenamento de datos entre o Estudio de Gravación e o de Animación

Descrición: O Estudio de Gravación deberá gardar os datos nunha cola, de onde os irá recollendo o Estudio de Animación para evitar a perda dos mesmos.

Relevancia: Alta

Restrición de deseño - RDe.05

Título: Cálculo da transformación entre os datos de calibración dos dous dispositivos

Descrición: O sistema deberá calcular, a partires dos datos de calibración, a transformación que é necesario aplicarlle aos datos de captura dos dispositivos para que traballen nun mesmo sistema de coordenadas.

Relevancia: Esencial

3.3. Requisitos funcionais

Os casos de uso son un bo punto de partida para a extracción de requisitos funcionais á hora de levar a cabo un desenvolvemento de software. Analizando minuciosamente os casos de uso é necesario atopar e concretar que funcionalidades son as que definen o software tal e como o cliente o entende. No caso do presente proxecto a especificación de requisitos extraida dos casos de uso é a seguinte:

Requisito funcional - RF.01

Título: Informar do número de dispositivos de captura conectados

Descrición: O sistema debe informar do número de dispositivos de captura que hai conectados ao equipo onde se executa.

Casos de uso relacionados: CU.01

Relevancia: Esencial

Requisito funcional - RF.02

Título: Obter datos de posición de ósos

Descrición: O sistema debe recoller datos de posición de ósos ao longo do tempo desde un dispositivo de captura de movemento sen marcas.

Casos de uso relacionados: CU.01

Relevancia: Alta

Requisito funcional - RF.03

Título: Obter de datos de rotación de ósos

Descrición: O sistema deberá obter datos de rotación de ósos, directamente ou mediante a transformación dos datos de posición en datos de rotación con respecto ao óso pai correspondente na xerarquía do esqueleto.

Casos de uso relacionados: CU.01

Relevancia: Esencial

Requisito funcional - RF.04

Título: Escoller o tipo de animación

Descrición: O sistema debe permitir que se escolla entre xerar unha animación baseada en posicións ou baseada en rotacións.

Casos de uso relacionados: CU.03, CU.04

Relevancia: Alta

Requisito funcional - RF.05

Título: Proporcionar acceso aos datos de captura

Descrición: O Estudio de Gravación deberá poder poñer a disposición do Estudio de Animación os datos de captura procedentes da gravación dunha sesión de interpretación.

Casos de uso relacionados: CU.01

Relevancia: Esencial

Requisito funcional - RF.06

Título: Adquirir os datos de captura

Descrición: O Estudio de Animación deberá poder recoller os datos que o Estudio de Gravación pon á súa disposición.

Casos de uso relacionados: CU.03

Relevancia: Esencial

Requisito funcional - RF.07

Título: Almacenar os datos de captura nun ficheiro

Descrición: O sistema debe permitir que se almacenen os datos de captura nun ficheiro cun formato acordado, unha vez rematada a sesión de interpretación.

Casos de uso relacionados: CU.02

Relevancia: Media

Requisito funcional - RF.08

Título: Cargar un ficheiro con datos de captura

Descrición: O sistema debe permitir que se carguen os datos de captura dunha sesión de interpretación desde un ficheiro.

Casos de uso relacionados: CU.04

Relevancia: Opcional

Requisito funcional - RF.09

Título: Obter datos de animación baseada en posicións

Descrición: O sistema debe permitir que se obteñan datos de animación baseada en posicións a partires dos datos de posición capturados, recibidos en tempo real ou desde ficheiro.

Casos de uso relacionados: CU.03, CU.04

Relevancia: Esencial

Requisito funcional - RF.10

Título: Obter datos de animación baseada en rotacións

Descrición: O sistema debe permitir que se obteñan datos de animación baseada en rotacións a partires dos datos de rotación capturados, recibidos en tempo real ou desde ficheiro.

Casos de uso relacionados: CU.03, CU.04

Relevancia: Esencial

Requisito funcional - RF.11

Título: Crear o esqueleto que aloxará a animación baseada en posicións

Descrición: O sistema debe permitir que se cree un esqueleto para almacenar os datos de animación baseada en posicións.

Casos de uso relacionados: CU.03, CU.04

Relevancia: Alta

Requisito funcional - RF.12

Título: Crear o esqueleto que aloxará a animación baseada en rotacións

Descrición: O sistema debe permitir que se cree un esqueleto para almacenar os datos de animación baseada en rotacións.

Casos de uso relacionados: CU.03, CU.04

Relevancia: Alta

Requisito funcional - RF.13

Título: Crear de maneira automática o esqueleto que aloxará a animación

Descrición: O sistema debe crear o esqueleto que aloxará a animación no caso de que o usuario non o teña creado previamente.

Casos de uso relacionados: CU.03, CU.04

Relevancia: Baixa

Requisito funcional - RF.14

Título: Transferir os datos de animación baseada en posicións ao esqueleto que a aloxará

Descrición: O Estudio de Gravación deberá poder tomar os datos xa transformados de animación baseada en posicións e transferilos á entidade que os aloxará no contorno do software de xeración de gráficos por ordenador.

Casos de uso relacionados: CU.03, CU.04

Relevancia: Esencial

Requisito funcional - RF.15

Título: Transferir os datos de animación baseada en rotacións ao esqueleto que a aloxará

Descrición: O Estudio de Gravación deberá poder tomar os datos xa transformados de animación baseada en rotacións e transferilos á entidade que os aloxará no contorno do software de xeración de gráficos por ordenador.

Casos de uso relacionados: CU.03, CU.04

Relevancia: Esencial

Requisito funcional - RF.16

Título: Informar se aínda non se calibrou no contorno de máis dun dispositivo de captura

Descrición: O sistema deberá informar que é necesaria a calibración para traballar con máis dun dispositivo de captura e mostrar un aviso no caso de que non se leve a cabo antes da gravación.

Casos de uso relacionados: CU.01

Relevancia: Esencial

Requisito funcional - RF.17

Título: Calibrar facendo uso do esqueleto completo no contorno con dous dispositivos de captura

Descrición: O sistema debe permitir que se calibre utilizando os datos recollidos da captura do esqueleto completo do intérprete durante certo tempo. Cando o usuario inicia a calibración, o sistema debe quedar en espera ata que o corpo completo do intérprete sexa perfectamente visible polos dous dispositivos a calibrar. O proceso de calibración captura os datos necesarios durante certo tempo. Cando remata, o sistema fai os cálculos necesarios para que os dous dispositivos traballen no mesmo sistema de coordenadas e informa ao usuario de que os dispositivos están calibrados.

Casos de uso relacionados: CU.01

Relevancia: Esencial

Requisito funcional - RF.18

Título: Procesar os datos de calibración no contorno de dous dispositivos de captura

Descrición: O Estudio de Gravación deberá procesar os datos recollidos polos dous dispositivos, facendo uso dos datos de calibración, para que a saída pertenza ao mesmo sistema de coordenadas.

Casos de uso relacionados: CU.01

Relevancia: Esencial

Requisito funcional - RF.19

Título: Elixir o fotograma de inicio da animación

Descrición: O sistema debe permitir modificar o fotograma de inicio da animación.

Casos de uso relacionados: CU.03, CU.04

Relevancia: Opcional

Requisito funcional - RF.20

Título: Desprazar no espazo a personaxe animada

Descrición: O sistema debe permitir que se escolla se a personaxe animada se despraza no espazo ou permanece no mesmo lugar durante o transcurso da animación.

Casos de uso relacionados: CU.03, CU.04

Relevancia: Opcional

Requisito funcional - RF.21

Título: Modificar a inclinación vertical dos dispositivos de captura

Descrición: O sistema deberá permitir que se modifique a inclinación vertical dos dispositivos de captura, sempre que estes o permitan.

Casos de uso relacionados: CU.01

Relevancia: Opcional

Requisito funcional - RF.22

Título: Capturar ordes de voz

Descrición: O sistema deberá permitir que se capturen e executen ordes de voz para o control da inclinación do dispositivo de captura, para o inicio da calibración e para o inicio e remate da gravación dunha sesión de interpretación.

Casos de uso relacionados: CU.01

Relevancia: Opcional

3.4. Requisitos non funcionais

Dentro dos requisitos non funcionais teñen cabida todas aquelas características que definen o proxecto pero que non se poden considerar funcionalidades.

3.4.1. Requisitos de dispoñibilidade

O requisitos de dispoñibilidade especifican que recursos son necesarios para poder levar a cabo o desenvolvemento do proxecto.

Requisito de dispoñibilidade - RDi.01

Título: Dous dispositivos de captura de movemento sen marcas

Descrición: Son necesarios dous dispositivos de captura de movemento sen marcas.

Relevancia: Esencial

Requisito de dispoñibilidade - RDi.02

Título: Ordenador con potencia gráfica

Descrición: É necesario para o desenvolvemento e probas do software.

Relevancia: Esencial

Requisito de dispoñibilidade - RDi.03

Título: Entorno de desenvolvemento de software (*IDE*)

Descrición: É necesario para o desenvolvemento do software. O *IDE* escollido dependerá dos requisitos en canto a dispositivo de captura, linguaxe e sistema operativo impostos.

Relevancia: Esencial

Requisito de dispoñibilidade - RDi.04

Título: Licencia educacional e instalación de *Autodesk Maya*

Descrición: É necesario para o desenvolvemento e probas da parte do sistema relacionada con *Autodesk Maya*.

Relevancia: Esencial

Requisito de dispoñibilidade - RDi.05

Título: Instalación de *Blender*

Descrición: É necesario para o desenvolvemento e probas da parte do sistema relacionada con *Blender*.

Relevancia: Esencial

Requisito de dispoñibilidade - RDi.06

Título: Estancia diáfana de mínimo 3x3 metros

Descrición: É necesario para as probas de gravación con dous dispositivos.

Relevancia: Esencial

3.4.2. Requisitos de proxecto

Requisito de Rendemento - RR.01

Título: Taxa de datos capturados superior a 24 fotogramas de captura por segundo

Descrición: O Estudio de Gravación debe poder adquirir e servir ao Estudio de Animación unha taxa de datos superior a 24 fotogramas de captura por segundo, para obter unha animación o máis fiel posible.

Relevancia: Esencial

Requisito de Almacenamento - RA.01

Título: Creación de ficheiro de captura de movemento de extensión `.kmc`

Descrición: O Estudio de Gravación deberá poder almacenar as sesións de interpretación realizadas nun ficheiro `.kmc` (*Kinect Motion Capture*).

Relevancia: Media

Requisito de Calendario - RCa.01

Título: Data límite para a entrega do TFG: Luns 7 de Xullo de 2014

Descrición: A data límite para a entrega do TFG será o luns 7 de Xullo de 2014 antes das 14 horas, que é a data establecida pola USC no regulamento de traballos fin de grao para o Grao en Enxeñaría Informática.

Relevancia: Esencial

3.4.3. Requisitos de calidade

Son aqueles requisitos que debe cumprir o proxecto para que a calidade final do produto sexa a esperada polo cliente.

Requisito de calidade - RQ.01

Título: Manual de usuario

Descrición: A aplicación deberá contar cun manual que facilite o uso da mesma ao usuario final.

Relevancia: Alta

Requisito de calidade - RQ.02

Título: Internacionalización da aplicación

Descrición: A aplicación deberá ser deseñada de xeito que poida ser adaptada para outras linguas e rexións sen que isto supoña cambios no deseño, e supoña cambios mínimos no código.

Relevancia: Opcional

Requisito de calidade - RQ.03

Título: Publicación con licencia de código aberto da aplicación e liberación do código da mesma

Descrición: Deberanse estudar as diferentes opcións en canto a licencias de código aberto para a publicación e liberación de código, e se todos os compoñentes e dependencias utilizadas permiten o uso de ditas licencias.

Relevancia: Opcional

3.4.4. Requisitos de interface de usuario

Nos requisitos de interface de usuario encádranse todas aquelas características que debe cumprir a interface do produto final para garantir unha experiencia de uso o máis sinxela e beneficiosa posible ao usuario.

Requisito de interface de usuario - RIU.01

Título: Representación dos datos de captura de maneira gráfica

Descrición: O sistema debe mostrar, en tempo real e de maneira gráfica, os datos de captura que están sendo recollidos durante a gravación da sesión de interpretación.

Relevancia: Alta

Requisito de interface de usuario - RIU.02

Título: Visualización da captura dos dous dispositivos para calibración

Descrición: No caso de dous dispositivos de captura, o sistema debe mostrar o que cada un dos dispositivos é capaz de captar, para que o usuario saiba durante a calibración cando está sendo visualizado por ámbolos dous ao mesmo tempo.

Relevancia: Alta

3.5. Matriz de trazabilidade

A matriz de trazabilidade proporciona unha idea global de como se distribúen as funcionalidades a desenvolver dentro dos casos de uso do produto.

		Casos de uso			
		CU.01	**CU.02**	**CU.03**	**CU.04**
Requisitos funcionais	**RF.01**	✓			
	RF.02	✓			
	RF.03	✓			
	RF.04			✓	✓
	RF.05	✓			
	RF.06		✓		
	RF.07	✓			
	RF.08				✓
	RF.09			✓	✓
	RF.10			✓	✓
	RF.11			✓	✓
	RF.12			✓	✓
	RF.13			✓	✓
	RF.14			✓	✓
	RF.15			✓	✓
	RF.16	✓			
	RF.17	✓			
	RF.18	✓			
	RF.19			✓	✓
	RF.20			✓	✓
	RF.21	✓			
	RF.22	✓			

Táboa 3.1: Matriz de trazabilidade

3.6. Enunciado do alcance do proxecto

3.6.1. Descrición do alcance do proxecto

O proxecto procura desenvolver un sistema cuxa finalidade sexa permitir a obtención dunha animación no contorno dun software de xeración de gráficos por ordenador a partires dos movementos dunha persoa captados por un dispositivo de captura de movemento de baixo custo. Esta animación debe poder obterse en tempo real mentres ten lugar a gravación dos movementos da persoa, ou ben a través dun ficheiro que conteña os datos de captura. Para evitar as oclusións que o intérprete pode ocasionar co seu propio corpo, o sistema debe permitir a captura de movemento facendo uso de dous dispositivos de captura, sendo necesaria unha calibración previa para que ambos dispositivos traballen como se fosen un só.

3.6.2. Criterios de aceptación do produto

Como a tecnoloxía proposta para o desenvolvemento é *Scrum* , a aceptación do produto levarase a cabo en diferentes fases ao longo de todo o proxecto. Organizaranse reunións co cliente ao final de cada *Sprint*, nas que se realizarán as probas propostas para o *Sprint* anterior, e só se dará por válido se o cliente da o visto bo. O feito de involucrar ao cliente e que vaia dando a súa opinión ao longo de todo o desenvolvemento da como resultado un produto final moito máis afín coas súas necesidades.

Con todo, o tempo de desenvolvemento do produto non debe exceder o acordado co cliente: **401,25 horas**.

3.6.3. Entregables do proxecto

Como se comentou no punto anterior, nas reunións periódicas que se realizan ao final de cada *Sprint*, entregarase ao cliente unha versión funcional, que inclúa os requisitos satisfeitos ata o momento. Desta maneira o cliente ten un produto que funciona aínda que estea inacabado, e pode comezar a traballar con el. O feito de que o cliente teña contacto co produto desde case o primeiro momento fará xurdir cambios nos requisitos, para adaptar moito mellor o produto ás necesidades do cliente.

Ao remate do proxecto o cliente recibirá, ademais do produto software acordado, o código fonte do mesmo e un manual de usuario. Axuntarase tamén a presente

memoria do traballo realizado e os procesos de enxeñaría seguidos, por petición expresa do cliente.

3.6.4. Exclusións do proxecto

Quedan excluídos do proxecto todos aqueles requisitos que non foran acordados no seu inicio e que pola súa complexidade poñan en risco o remate en tempo e forma do mesmo. Aínda que *Scrum* está pensado para que existan modificacións na especificación de requisitos ao longo de todo o desenvolvemento, os cambios que se realicen non deben desviarse en exceso do alcance acordado.

3.6.5. Restricións do proxecto

Desenvolverase o sistema no contorno de dous paquetes software de xeración de gráficos por ordenador, un comercial e outro de código aberto. O criterio seguido para esta elección en concreto é un compromiso entre acadar un grande número de usuarios e o coñecemento previo de ambos paquetes por parte dos desenvolvedores do proxecto:

- A elección no caso do software comercial é *Autodesk Maya* [21], unha aplicación cun prezo dos máis elevados do mercado, pero que é un dos paquetes software deste tipo con máis anos de experiencia e máis utilizados na industria tanto para a produción de cine como para a creación de videoxogos.

- No caso do software non comercial a elección é *Blender* [22], que é o paquete software gratuíto de código aberto máis potente e con maior número de usuarios.

3.6.6. Supostos do proxecto

O equipo de desenvolvemento da por suposto que se contará cos dispositivos de baixo custo necesarios para levar a cabo o proxecto, desde o inicio deste.

3.7. Organización do traballo

Como xa se explicou anteriormente, **Scrum** é unha metodoloxía áxil que pretende involucrar ao cliente durante todo o desenvolvemento do proxecto. Para isto,

o traballo divídese en iteracións denominadas **Sprints**, para as que se deseñarán probas que deberán ser validadas polo cliente antes de comezar o seguinte *Sprint*.

Nun proxecto como este, que conta cun nivel de incerteza elevado, acordouse comezar polo que en *Scrum* se denomina *Sprint 0*. O **Sprint 0** abarca todas as tarefas necesarias que debe levar a cabo o equipo de desenvolvemento para poder comezar co traballo con solvencia e determinación. Este *Sprint 0* non implica que o equipo non se poida documentar e formar máis adiante, durante o transcurso do desenvolvemento, pero proporciónalle máis criterio á hora de avaliar a dificultade das tarefas, de priorizar entre unhas e outras, e de estimar o tempo que levará cada unha.

Para definir os *Sprints* nos que se dividirá o traballo, incluído o *Sprint 0*, asignaráselle a cada un deles un nome, unha descrición e a duración do traballo realizado. Para contextualizalos dentro do proxecto indicarase en que fase da planificación temporal se encadran. Para aqueles *Sprints* que impliquen traballo sobre o produto para levar a cabo funcionalidades acordadas, indicarase que requisitos foron satisfeitos.

A duración dos *Sprints* reflectirase en horas, tendo en conta que o proxecto se levará a cabo a un ritmo de 20 horas de traballo por semana.

Sprint 0

Título: Preparación para o proxecto

Descrición:

- Enunciado do alcance, obxectivos do proxecto e descrición do produto.
- Estudo do estado da técnica.
- Avaliación das diferentes tecnoloxías que permitan levar a cabo o proxecto.
- Formación inicial: Busca e primeira análise da documentación de cada unha das posibles tecnoloxías: libros, publicacións e recursos web.
- Elección da metodoloxía de desenvolvemento.

Duración: 29 horas

Fase(s): Xestión do proxecto e Preparación

Sprint - Sp.01

Título: Análise de requisitos

Descrición: Identificación de casos de uso, extracción e especificación inicial de requisitos e definición do alcance.

Duración: 15.5 horas

Fase(s): Xestión do proxecto e Iniciación

Sprint - Sp.02

Título: Análise de tecnoloxías e ferramentas

Descrición: Análise das diferentes tecnoloxías involucradas, das ferramentas necesarias e definición do laboratorio de traballo.

Duración: 6 horas

Fase(s): Xestión do proxecto, Preparación e Iniciación

Sprint - Sp.03

Título: Prototipo vertical/Proba de concepto

Descrición: Creación dun **prototipo vertical** ou **proba de concepto** para estudar a viabilidade do proxecto.

Duración: 20 horas

Fase(s): Preparación

Sprint - Sp.04

Título: Deseño preliminar

Descrición: Definición da arquitectura do sistema. Deseño do subsistema Servidor. Deseño do subsistema Cliente.

Duración: 18 horas

Requisitos involucrados: RF.01

Fase(s): Xestión do proxecto e Iniciación

Sprint - Sp.05

Título: Control de conexión dos dispositivos de captura

Descrición: Primeiros pasos na implementación do sistema, comezando coa elaboración do módulo de control de conexión de dispositivo(s) de captura de movemento ao equipo.

Duración: 18.5 horas

Requisitos involucrados: RF.01

Fase(s): Desenvolvemento

Sprint - Sp.06

Título: Adquisición e mostra de datos de posición

Descrición: Desenvolvemento dos módulos de adquisición e mostra dos datos de posición no sistema. Implementación dunha cola para almacenar os datos de captura.

Duración: 21.5 horas

Requisitos involucrados: RF.02, RIU.01, RDe.04

Fase(s): Desenvolvemento

Sprint - Sp.07

Título: Conexión con *Autodesk Maya*

Descrición: Primeiros pasos no desenvolvemento do sistema para *Autodesk Maya*. Desenvolvemento e integración do módulo que define e controla a conexión con Maya.

Duración: 19.5 horas

Requisitos involucrados: RF.05, RF.06, RDe.01

Fase(s): Desenvolvemento

Sprint - Sp.08

Título: Animación baseada en posicións en *Autodesk Maya*

Descrición: Desenvolvemento do módulo para crear un esqueleto base que almacenará os datos de posición. Desenvolvemento do módulo para xerar unha animación baseada en posicións a partires dos datos de captura.

Duración: 24.25 horas

Requisitos involucrados: RF.08, RF.10, RF.12, RF.13, RDe.01

Fase(s): Xestión do proxecto e Desenvolvemento

Sprint - Sp.09

Título: Animación baseada en rotacións en *Autodesk Maya*

Descrición: Adquisición de datos de rotación desde o(s) dispositivo(s) de captura. Módulo de creación do esqueleto base que almacenará os datos de rotación. Módulo de xeración de animación baseada en rotacións a partires destes datos. Integración dos diferentes módulos dentro de *Autodesk Maya*.

Duración: 22 horas

Requisitos involucrados: RF.03, RF.09, RF.11, RF.12, RF.14, RDe.01

Fase(s): Desenvolvemento

Sprint - Sp.10

Título: Conexión con *Blender*

Descrición: Primeiros pasos no desenvolvemento do sistema para *Blender*. Desenvolvemento e integración do módulo que define e controla a conexión con *Blender*.

Duración: 22.25 horas

Requisitos involucrados: RF.05, RF.06, RDe.02

Fase(s): Desenvolvemento

Sprint - Sp.11

Título: Animación baseada en rotacións en *Blender*

Descrición: Módulo de creación do esqueleto base que almacenará os datos de rotación. Módulo de xeración de animación baseada en rotacións a partires destes datos. Integración dos diferentes módulos dentro de *Blender*.

Duración: 12 horas

Requisitos involucrados: RF.09, RF.11, RF.12, RF.14, RDe.02

Fase(s): Desenvolvemento

Sprint - Sp.12

Título: Calibración I - Adquisición de datos para calibración

Descrición: Comezo do desenvolvemento para o contorno de dous dispositivos de captura do módulo de calibración. Desenvolvemento da parte do módulo de calibración que captura datos de posicións dos ósos do esqueleto desde os dous dispositivos ao mesmo tempo. Desenvolvemento da parte do módulo de calibración que mostra a representación gráfica dos datos do esqueleto capturado nos dous dispositivos.

Duración: 23 horas

Requisitos involucrados: RF.15, RF.16, RIU.01

Fase(s): Xestión do proxecto e Desenvolvemento

Sprint - Sp.13

Título: Calibración II - Cálculo dos parámetros da transformación afín

Descrición: Desenvolvemento da parte do módulo de calibración que calcula os parámetros da transformación afín que é necesario aplicarlle aos datos dos dispositivos para que traballen nun mesmo sistema de coordenadas.

Duración: 31 horas

Requisitos involucrados: RDe.05

Fase(s): Desenvolvemento

Sprint - Sp.14

Título: Calibración III - Procesamento dos datos de captura

Descrición: Desenvolvemento da parte do módulo de adquisición de datos de captura que procesa os datos dos dous dispositivos para que traballen nun mesmo sistema de coordenadas.

Duración: 22.5 horas

Requisitos involucrados: RF.17

Fase(s): Desenvolvemento

Sprint - Sp.15

Título: Ficheiro de datos de captura

Descrición: Desenvolvemento de parte do módulo de posicionamento de datos que almacenará os datos de captura dunha sesión de interpretación nun ficheiro en texto plano. Desenvolvemento do módulo que le os datos de captura dun ficheiro en texto plano e os interpreta para xerar unha animación.

Duración: 10.5 horas

Requisitos involucrados: RF.07, RF.08, RDe.03, RA.01

Fase(s): Desenvolvemento

Sprint - Sp.16

Título: Remate do sistema

Descrición: Desenvolvemento do módulo de ordes de voz. Desenvolvemento do módulo que xestiona a inclinación vertical dos dispositivos. Implementación da parte do módulo de adquisición de datos que permite definir o fotograma no que se inicia a animación. Implementación da parte do módulo de xeración de animación que permite escoller se a personaxe se desprazará no espazo durante a animación.

Duración: 20.5 horas

Requisitos involucrados: RF.19, RF.20, RF.21, RF.22

Fase(s): Xestión do proxecto e Desenvolvemento

Sprint - Sp.17

Título: Manual de usuario

Descrición: Elaboración do manual de usuario.

Duración: 5 horas

Requisitos involucrados: RQ.01

Fase(s): Xestión do proxecto

Sprint - Sp.18

Título: Conclusión do proxecto

Descrición: Validación e probas co sistema completamente integrado. Revisión e remate da memoria do proxecto.

Duración: 14 horas

Fase(s): Xestión do proxecto e Conclusión

Este conxunto de *Sprints* non representa unha planificación, senón o resultado de ter aplicado *Scrum* para a realización deste proxecto. O catálogo de requisitos e por extensión o conxunto de tarefas que había que levar a cabo foi evolucionando ao longo do proxecto. Determinadas tarefas estaban claras desde o inicio e foron as que marcaron o camiño a seguir, pero outras menos relevantes fóronse adaptando a este camiño segundo se ía avanzando.

A duración dos *Sprints* oscila entre as vinte e as corenta horas. Se temos en conta que o tempo de traballo por semana estaba estipulado en 20 horas, o resultado é un tempo por *Sprint* de entre unha e dúas semanas. A diferencia de tempo entre uns e outros *Sprints* débese á grande incerteza coa que se iniciou este proxecto, e a que certas tarefas resultaron ser máis complexas de levar a cabo do que nun principio se esperaba. De todas maneiras, o desfase entre *Sprints* non é tan grande e pódese dicir que a aplicación desta metodoloxía nun proxecto destas características proporcionou un resultado satisfactorio.

Capítulo 4

Análise de tecnoloxías e ferramentas

Neste capítulo avaliaranse as tecnoloxías e ferramentas necesarias para acadar os obxectivos do proxecto, e dentro de cada unha, as posibilidades que foron consideradas. É necesario salientar que a maioría destas tecnoloxías están relacionadas, de tal maneira que unha determinada elección nunha delas pode levar consigo restricións importantes noutras. Por esta razón, avaliarase por separado cada tecnoloxía e as súas alternativas, para rematar cun apartado no que se explican e xustifican as decisións tomadas.

4.1. Laboratorio de traballo

Para a realización deste proxecto é necesario, en primeiro termo, definir cales serán as tecnoloxías e ferramentas que conformarán o contorno de traballo básico. A razón de tomalas en consideración de maneira conxunta é pola estreita relación que existe entre elas. Este contorno de traballo básico estará formado polos tres puntos esenciais para a consecución dos obxectivos do proxecto:

- O modelo do dispositivo de captura de movemento sen marcas de baixo custo.
- A libraría *SDK* ou *API* que se utilizará para interactuar co dispositivo.
- A linguaxe de programación na que se desenvolverá.

A continuación analizaranse as diferentes opcións consideradas para cada un destes puntos.

4.1.1. Dispositivo de captura de movemento sen marcas de baixo custo

Nos últimos anos da pasada década, *Microsoft* entrou en negociacións coa empresa *PrimeSense* [29] en relación cunha tecnoloxía de baixo custo que estaba a desenvolver para a captura de movemento sen marcas. *Microsoft* mercou esta tecnoloxía coa única idea de competir no mundo dos videoxogos cos novos tipos de interacción *usuario-videoxogo* que as compañías rivais estaban sacando ao mercado, como pode ser o *WiiMote* [30] da empresa *Nintendo* ou o *PlayStation Eye* [31] da empresa *Sony*. Finalmente, en 2010, *Microsoft* sacou o *Kinect* como periférico para a súa consola *XBOX360*, co que se podía interactuar cos videoxogos e a consola sen necesidade dun mando. Dous anos despois *PrimeSense*, a empresa pioneira nesta tecnoloxía, aliouse con *Asus* para sacar ao mercado a gama *Xtion* [32], consistente

nunha serie de dispositivos con características e prezos similares a *Kinect*, pero exclusivamente para *PC*.

4.1.2. Librarías de conexión co dispositivo de captura

Cando o *Kinect* de *Microsoft* saíu ao mercado en 2010, non tardaron en xurdir intentos de conectar este dispositivo a un ordenador persoal, e tratar de aproveitar ao máximo esta nova tecnoloxía.

Un destes intentos tiña por detrás unha nova organización chamada *OpenNI* [33], creada ao mesmo tempo que o *Kinect* saía ao mercado, e que contaba nas súas filas como membro á propia *PrimeSense*, a empresa que deseñou o *Kinect*. Esta organización sacou os primeiros *drivers* de código aberto para *Kinect* xunto cun *middleware* chamado *Nite* para a captura de movemento. Todo isto conforma unha *API* de código aberto chamada *OpenNI Framework* [33] que permite acceder a todos os recursos do *Kinect*, como recoñecemento de voz, recoñecemento de xestos e recoñecemento de movemento en bípedes. O proxecto continúa activo, con actualizacións cada certo tempo, e conta con moito éxito na comunidade de desenvolvedores. Como foi a primeira alternativa que xurdiu para poder interactuar co *Kinect* desde un ordenador, existe bibliografía interesante como axuda e soporte á hora de abordar un proxecto con esta libraría.

Pola súa banda, *Microsoft*, observando a reacción dos usuarios cos *drivers* de *OpenNI*, decidiu liberar os seus propios *drivers* e o *SDK* de desenvolvemento para *Kinect*, e incluso sacar ao mercado unha versión de *Kinect* para desenvolvedores. Este proxecto continúa tamén moi activo, con actualizacións cada pouco tempo. Pero onde destaca este proxecto é na documentación, xa que ademais de contar con bastante bibliografía interesante, os desenvolvedores contan cunha axuda moi detallada no sitio web de *Microsoft* de todas as partes que conforman dita *API*.

A última alternativa considerada foi *FAAST* [34], un proxecto encadrado no *Institute for Creative Technologies da University of Southern California*. O *middleware FAAST* é probablemente o máis completo dos tres, xa que permite traballar cos *drivers* de *Microsoft* e cos de *OpenNI*. Por outra banda, aínda que a última versión de *FAAST* saíu en Decembro de 2013, o seu desenvolvedor avisa que existe a posibilidade de que o proxecto se deteña por falta de tempo. Ademais, a documenta-

ción atopada sobre o desenvolvemento con este *middleware* é escasa en comparación coa de *OpenNI* ou o *SDK* de *Microsoft*.

4.1.3. Linguaxe de desenvolvemento

A libraría de recollida de datos escollida será a que restrinxa a linguaxe ou linguaxes nas que será posible levar a cabo o desenvolvemento. No caso de *OpenNI*, aínda que existen *wrappers* tanto de *Java* como de *Python*, a linguaxe de desenvolvemento nativa é, da mesma maneira que en *FAAST*, *C++*. O *SDK* de *Microsoft* está concibido para que o desenvolvemento se realice no *framework .NET*, dando a posibilidade aos desenvolvedores de realizar a programación en *C++*, *C#* ou *VB.NET*. Este último tamén conta cunha *wrapper* en *Python* con acceso a moitas das funcionalidades.

Por outro lado, os paquetes software escollidos para o proxecto, *Autodesk Maya* e *Blender*, brindan a posibilidade de desenvolver novas funcionalidades de dúas maneiras diferentes: a primeira, como un *plugin*, onde o desenvolvemento se realizaría en C++ accedendo directamente á *API*, e o resultado sería un ficheiro compilado que o paquete software carga a demanda do usuario. A segunda opción, como un *script*, onde o desenvolvemento se realiza nunha linguaxe de *scripting*, e o resultado é un ficheiro en texto plano que o paquete software le e interpreta a demanda do usuario no momento da chamada. Aínda que os *plugins* presentan un mellor rendemento polo feito de ser compilados, unha linguaxe de *scripting* como *Python*, soportada polos dous paquetes software elixidos e que traballa directamente contra a *API* correspondente, proporciona un rendemento máis que aceptable na maioría dos casos. No caso de *Autodesk Maya* existe unha segunda alternativa de linguaxe de *scripting* coñecido como *MEL(Maya Embedded Language)*. Esta alternativa traballa nun nivel de abstracción máis alto que *Python*, provocando que o rendemento dos *scripts* desenvolvidos nesta linguaxe se resinta de maneira notable.

4.1.4. Definición do laboratorio de traballo

Ao inicio dun proxecto o descoñecemento das tecnoloxías que van ser utilizadas para cumprir cos obxectivos por parte de quen o vai desenvolver produce moita incerteza. Por esta razón, o feito de que o desenvolvedor coñeza e xa teña traballado co *framework .NET* e, máis concretamente, coa linguaxe **C#**, erixe a esta linguaxe como a preferida para levar a cabo o desenvolvemento da parte do Estudio de Gravación. Da mesma forma, o desenvolvedor coñece e tivo oportunidade de realizar

algunhas implementacións con **Python**, polo que a integración do sistema dentro do contorno do software de xeración de gráficos por ordenador realizarase nesta linguaxe na medida do posible. Estas dúas eleccións levan consigo un aforro en tempo de formación noutras linguaxes, coas que o desenvolvedor tivo moito menos contacto.

A escolla de *C#* para o desenvolvemento da parte do Estudio de Gravación restrinxe as opcións en canto ao dispositivo de captura e en canto a libraría de recollida de datos. O par **Dispositivo *Kinect* - *Kinect for Windows SDK*** é a única que permite o desenvolvemento de aplicacións utilizando *C#* e polo tanto é a escollida para levar a cabo esta parte do sistema.

En canto a dispositivos de captura de movemento sen marcas de baixo custo, o **Kinect** é o máis estendido; ten un prezo similar ao que ofrecen as alternativas e é o único que permite a utilización do *SDK* publicado por *Microsoft*. Este *SDK* que *Microsoft* puxo a disposición dos desenvolvedores ten un mantemento continuo, con actualizacións cada pouco tempo, sempre con melloras, acceso a novas funcionalidades e ademais, dentro das alternativas antes comentadas, é a que conta cunha documentación máis extensa e accesible.

Funcionamento do Kinect

O *Kinect* conta cunha serie de tecnoloxías que lle permiten capturar datos de imaxe e son nun determinado entorno ou estancia. Ademais conta cunha tecnoloxía baseada en luz infravermella que lle permite calcular a profundidade. A continuación detállanse as diferentes partes que conforman o *Kinect* (Fig. 4.1) e cal é a súa función:

Na parte máis á esquerda está o emisor de luz infravermella; ao seu lado atópase o indicador *LED* que se acende ao conectar o aparato; a continuación atópase a cámara *RGB*, e finalmente na parte dereita sitúase o sensor infravermello.

Para detectar a profundidade o dispositivo emprega o emisor e o sensor de infravermello de forma conxunta. O emisor proxecta de forma continua sobre todos os obxectos da escena puntos de luz infravermella nun patrón semialeatorio. Estes puntos non son visibles para o ser humano, pero si para o sensor de infravermellos co que conta o *Kinect*. Cando a luz infrarroxa emitida atopa un obxecto, é reflectida de volta cara o dispositivo, que coa cámara de infrarroxo é capaz de medir a distancia desde a que foi reflectido ese punto de luz. Desta forma o dispositivo é capaz de convertir os datos que recibe en información de profundidade.

Na base do dispositivo atópase un pequeno motor que permite inclinar a cámara 30 graos cara arriba ou cara abaixo.

Figura 4.1: Imaxe que mostra as diferentes partes que conforman o *Kinect* de *Microsoft*.

Por último, na súa parte inferior atópanse catro micrófonos que traballan combinados entre sí para filtrar o ruido de fondo e detectar a posición relativa dunha persoa que fala dentro da estancia [36].

4.2. Sistema operativo

A decisión en canto á libraría de captura de datos e á linguaxe de implementación restrinxen as posibilidades en canto ao sistema operativo de traballo unicamente a *Microsoft Windows*.

4.3. IDE de desenvolvemento

A decisión na linguaxe de implementación deixa poucas opcións para levar a cabo o desenvolvemento. A opción máis viable e sinxela para desenvolver para o *framework .NET* en *Windows* é **Visual Studio**. Existe unha versión gratuíta do seu *IDE* coñecida como **Visual Studio Express** que *Microsoft* pon a disposición de estudantes ou desenvolvedores interesados en crear software sen ánimo de lucro, e que será a que se utilice para o desenvolvemento da parte do Estudio de Gravación.

4.4. Interface gráfica

A parte do Estudio de Gravación do sistema será unha aplicación de escritorio que permitirá ao usuario interactuar cos dispositivos de captura, e que á súa vez será a que se comunique coa parte do Estudio de Animación no caso de querer realizar unha gravación dunha sesión de interpretación nun dos paquetes software propostos. Como existe a posibilidade de que non haxa conexión con ningún paquete software e que o usuario só queira gardar a sesión nun ficheiro, a aplicación debería mostrar en todo momento a interpretación que está levando a cabo o actor plasmada nun avatar nun contorno 3D.
As alternativas para realizar interfaces gráficas en *.NET*, e concretamente en C# son dúas:

- A primeira e máis coñecida é **WinForms**, que proporciona unha libraría para construír unha aplicación de escritorio de maneira rápida e sinxela. Todos os compoñentes gráficos son clases, polo que a interacción con eles a nivel programático é moi intuitiva.

- Por outra banda, existe a tecnoloxía coñecida como **WPF** *(Windows Presentation Foundation)*, que reinventa o deseño da parte gráfica para que esta se enmarque a medio camiño entre as aplicacións de *Windows* e as aplicacións web. *WPF* aporta unha eficiencia e unha potencia gráfica superior a *WinForms*, contando entre as súas características coa posibilidade de mostrar gráficos en 3D.

O feito de que WPF proporcione un rendemento superior en aplicacións *multimedia*, e sobre todo que permita a posibilidade de mostrar gráficos en 3D, fan que esta sexa a candidata elixida como linguaxe de implementación da interface gráfica.

4.5. Comunicación entre partes do sistema

Unha parte de grande relevancia e criticidade dentro do sistema é a comunicación entre a parte denominada *Estudio de Gravación*, encargada de recoller os datos proporcionados polos dispositivos de captura e compartilos, debidamente formateados, e a parte denominada *Estudio de Animación*, que se encarga de xerar a animación dentro do paquete software correspondente. Debe ser unha comunicación fluída, sen perdas de datos, e cunha taxa de transferencia alta.

Aínda que existen alternativas para a comunicación entre diferentes aplicacións, sempre van depender de moitos factores, sendo o máis relevante a linguaxe de desenvolvemento utilizada. Para evitar restricións impostas por decisións tomadas nesta cuestión, a tecnoloxía elixida para levar a cabo a comunicación son os *sockets*, concretamente **sockets TCP**. Existen varias razóns para esta decisión: a primeira é porque calquera das linguaxes comentadas no punto anterior soporta a utilización de *sockets*, a excepción de *MEL*; polo tanto sexa cal sexa a escolla tanto para a parte do Estudio de Gravación como para a do Estudio de Animación a compatibilidade non vai ser un problema. Ademais, isto asegura a independencia con respecto á implementación na parte de comunicación, acadando así o mínimo acoplamento entre partes. A segunda é que o feito de traballar a máis baixo nivel, sen ningún *middleware*, asegura un rendemento máis alto no transporte de datos. A terceira e última razón é que o feito de utilizar *sockets TCP*, que son orientados a conexión, asegura a integridade nos datos recibidos pola parte do Estudio de Animación.

Capítulo 5

Deseño de software

Neste capítulo abórdase en detalle o deseño do sistema que é necesario desenvolver. En primeiro lugar establécese unha arquitectura, marcando as pautas de como vai estar organizado o sistema a mais alto nivel. Tomando a arquitectura como punto de partida, defínense detalladamente cada unha das súas partes tanto de maneira estática co deseño de clases como de maneira dinámica con diagramas de secuencia dos diferentes casos de uso.

5.1. Arquitectura do sistema

Na elección da arquitectura mais adecuada para un sistema como o que aquí se propón é necesario partir da análise das necesidades do cliente. Se esta análise é o suficientemente detallada e robusta pódense extraer de maneira sinxela as partes que deben conformar o sistema a mais alto nivel, tendo sempre moi presentes as necesidades do cliente e a maneira na que se vai usar o produto.

No proxecto que nos ocupa, segundo o extraido das conversas co cliente, o sistema debe realizar dúas tarefas clave que se sitúan ao nivel mais alto:

- Adquisición e tratamento de datos desde o(s) dispositivo(s) de captura
- Xeración dunha animación a partires dos datos de captura

Como se pode observar, aínda que as partes que conforman o sistema están ben diferenciadas existe unha clara dependencia entre elas: a parte de adquisición recolle os datos do(s) dispositivo(s) de captura e debe enviarllos á parte de xeración de animación para que esta realice a súa función. Esta dependencia ten a súa concreción de maneira directa nunha arquitectura **Cliente-Servidor** (Fig. 5.1).

Nesta orde de cousas podemos renomear ás partes do sistema como **subsistemas** e describilas da seguinte maneira:

- **Subsistema Servidor** - *Estudio de Gravación*: Adquisición e tratamento de datos desde o(s) dispositivo(s) de captura
- **Subsistema Cliente** - *Estudio de Animación*: Xeración dunha animación a partires dos datos de captura

A continuación descríbense en detalle a parte Servidor, a parte Cliente e a maneira na que estas partes se comunican.

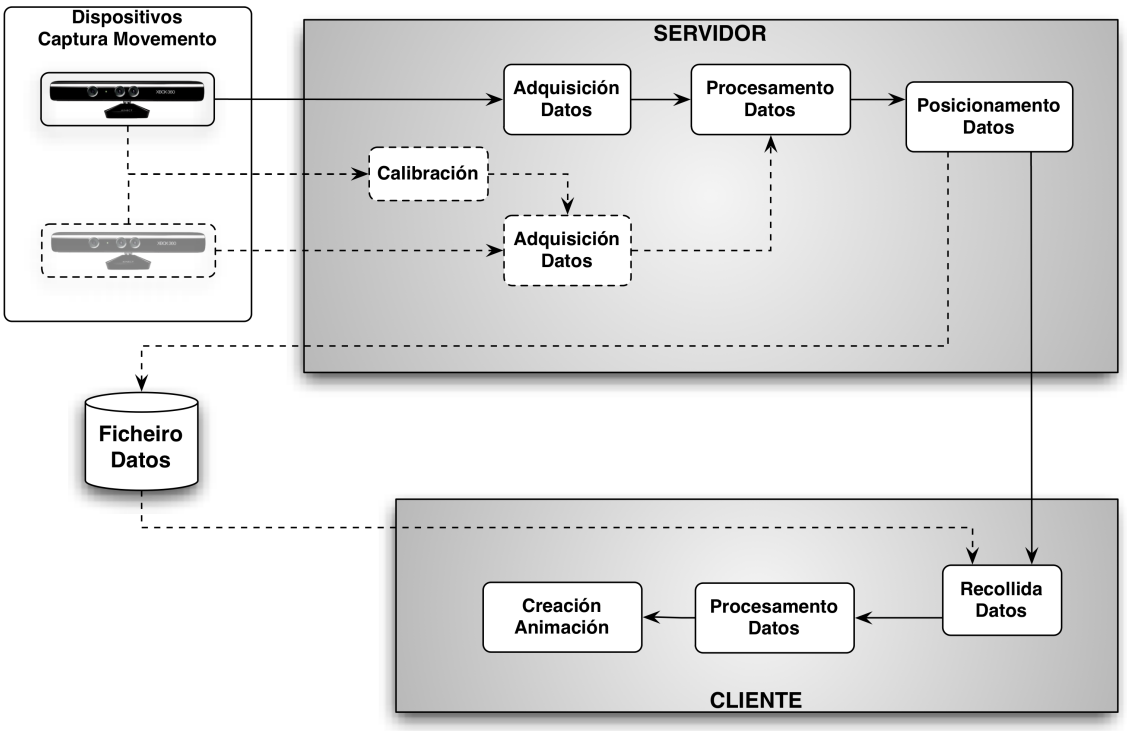

Figura 5.1: Arquitectura do sistema

5.2. Servidor

Como se indicou na sección anterior, o subsistema Servidor é o encargado de interactuar cos dispositivos de captura. A interacción con elementos *hardware*, e sobre todo, a adquisición de datos desde estes en tempo real, require un esforzo importante na planificación. É necesario definir claramente que partes do desenvolvemento van ter interacción directa cos dispositivos externos e de que maneira. Tendo isto en conta, unha boa práctica para levar a cabo o desenvolvemento do subsistema Servidor é dividilo en partes mais pequenas segundo a **responsabilidade** que teñen. Esta separación reduce o acoplamento, permitindo ao equipo de desenvolvedores traballar por separado con cada parte dentro do Servidor, facendo mais áxil o desenvolvemento. Ademais como cada parte ten unha interacción moi concreta cos dispositivos, esta división facilita e fai mais fluída a depuración de código.

Para definir de maneira detallada o deseño do subsistema Servidor é unha boa práctica comezar especificando un patrón de arquitectura no que se indicará como se organizan as responsabilidades dentro deste subsistema. A continuación, mostrase a

división desta arquitectura en clases, tanto a nivel xeral do Servidor como a nivel de cada responsabilidade. E por último, detállase como é a interacción co subsistema Servidor en forma de diagramas de secuencia daqueles casos de uso nos que esta parte está involucrada.

5.2.1. Patrón de arquitectura

Un patrón de arquitectura proporciona un esquema de organización estrutural para un desenvolvemento de software, dividindo o sistema en partes segundo a súa responsabilidade e propoñendo unha serie de interrelacións e restricións entre ditas partes. A elección dun patrón arquitectónico ten moito peso no proxecto e debe escollerse con criterio. Un cambio de patrón de arquitectura en estadios avanzados dun desenvolvemento supón en moitos casos, unha perda importante de tempo e cartos.

O patrón de arquitectura máis estendido e máis utilizado a día de hoxe é o ***MVC* (Model-View-Controller).** Neste patrón, a vista, que é coa que interactúa directamente o usuario, utiliza o controlador como o seu asistente, enviándolle os comandos ou ordes segundo as accións do usuario, e á súa vez o controlador realiza os cambios pertinentes no Modelo. A Vista é actualizada co novo estado dos datos cada vez que existe un cambio no modelo. A dependencia que existe entre Modelo e Vista leva consigo un certo acoplamento que pode resultar problemático no caso de ser necesario algún cambio importante nalgunha destas partes.

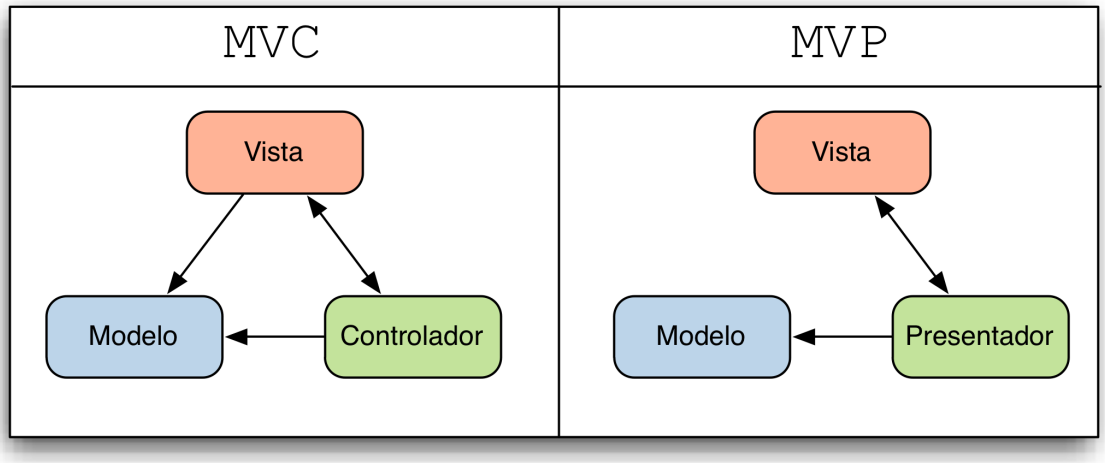

Figura 5.2: Comparación *MVC* - *MVP*

Unha alternativa a *MVC* é o ***MVP* (Model-View-Presenter)** [35]. *MVP* é un patrón de arquitectura para o desenvolvemento de software que nace como unha evolución de *MVC*. A diferencia entre ambos radica basicamente nas responsabilidades de cada unha das partes. En *MVP*, Modelo e Vista son completamente independentes, e non teñen contacto directo, senón que se comunican a través do Presentador. Desta maneira, toda a lóxica de negocio e a lóxica de presentación será asumida polo Presentador. A Vista pola súa parte é totalmente pasiva, poñéndose ao servizo do Presentador, que será o que indique que se mostra, cando se mostra e como se mostra. En canto ao Modelo, proporciona unha interface para definir os datos cos que o Presentador vai traballar. Como se pode observar, o Presentador é o que mantén unida toda a estrutura. Se fose necesario realizar cambios na Vista ou no Modelo, as modificacións so chegarían ao Presentador deixando intacta a terceira parte.

Este baixo acoplamento entre as partes permite cambios radicais na Vista, que afectarían moi lixeiramente ao resto do desenvolvemento. Nun proxecto con tanta incerteza, que se espera que vaia sufrir cambios e evolucionar moito durante o desenvolvemento é interesante contar co nivel de flexibilidade que proporciona este baixo acoplamento.

O patrón de arquitectura escollido para levar a cabo o desenvolvemento do subsistema Servidor do sistema é MVP. Cada responsabilidade estará representada por unha parella Vista-Presentador.

Tendo en conta estas responsabilidades e a organización proposta, tomouse a decisión de extraer certa lóxica dos Presentadores, que non é exclusiva da responsabilidade que representan, e crear con ela un conxunto de clases que non precisan manter o estado e que denominaremos **Helpers**.

5.2.2. Deseño e implementación

Nesta sección abordarase a organización en clases do subsistema Servidor dentro da arquitectura *MVP*. Para distinguir máis claramente en que parte desta arquitectura se encadra cada clase utilizase un código de cores que se manterá durante o resto de diagramas da parte Servidor.

O diagrama de clases que se mostra na figura 5.3 pretende dar unha visión xeral de como se organizan as clases dentro da arquitectura proposta. Convén aclarar que non se mostran nin os atributos nin os métodos das clases involucradas para ofrecer unha mellor visualización global. Neste diagrama apréciase claramente como a comunicación entre a *Vista* e o *Modelo* é a través do *Presentador*. Cada parella *Vista-Presentador* que se pode ver neste diagrama representa unha responsabilidade diferente dentro do conxunto do subsistema Servidor, permitindo a súa reutilización por separado en caso de que fose necesario.

As responsabilidades nas que se divide o subsistema Servidor son as seguintes:

- Xestión de datos/*Data Address*
- Calibración de dispositivos/*Device Calibration*
- Recollida de datos/*Data Gathering*
- Recoñecemento de voz/*Speech Recognition*
- Coordinación do subsistema Servidor

A continuación detállase cada unha destas responsabilidades por separado mostrando, desta vez, as clases involucradas con todos os elementos que as definen. Nestes apartados, xunto coas tarefas que desempeña cada responsabilidade, explicaranse aqueles detalles de implementación que se consideran salientables para un mellor entendemento do funcionamento do módulo.

Responsabilidade: Xestión de datos - *Data Address*

O módulo de *Xestión de datos* do subsistema Servidor é o encargado de establecer e xestionar a conexión co subsistema Cliente no caso de que este o solicite, e do envío de datos a este, de almacenar os datos de captura durante a gravación dunha sesión de interpretación, e de crear un ficheiro cos datos da sesión, se así o desexa o usuario, ao remate da gravación. O diagrama de clases que representa esta responsabilidade móstrase na figura 5.4.

Para a conexión co subsistema Cliente utilízanse **sockets TCP**, como xa se comentou anteriormente. Estes *sockets* son orientados a conexión para evitar a perda de datos. Como o subsistema Servidor ten entidade e funcionalidade sen conexión cun cliente, optouse por utilizar *sockets* **asíncronos** que unha vez activados, esperan de maneira pasiva a que un cliente se conecte. Cando isto sucede a conexión queda

establecida, pero no caso de que non se conecte un cliente, o *socket* non queda activamente buscando conexións durante toda a execución, como pasaría no caso dos síncronos.

Para almacenar os datos durante a gravación dunha sesión de interpretación este módulo utiliza unha cola, tal como se especifica na restricción de deseño RDe.04. A elección dunha cola como estrutura de datos para esta tarefa é debida a que o cliente expresou o desexo de que non se deberían perder datos de captura durante a gravación dunha sesión de interpretación. Así, este módulo almacena nesta cola os datos de captura dunha sesión de interpretación en tempo real mentres esta está tendo lugar. Se un cliente se conecta ao subsistema Servidor, este comeza a enviarlle os datos almacenados na cola comezando polo primeiro que entrou, ata baleirar a cola. A partires dese momento a cola actúa de ponte entre o módulo que recolle os datos de captura e o cliente, xa que cando un dato entra é enviado de seguido a este cliente.

Por último, os datos de captura que van sendo almacenados na cola tamén son gardados nun documento en memoria. Desta maneira, se o cliente o solicita ao final dunha sesión de interpretación, estes datos son transferidos a un ficheiro da súa elección.

Responsabilidade: Calibración de dispositivos - *Device Calibration*

O módulo de *Calibración de dispositivos* só entra en xogo no caso de que haxa máis dun dispositivo de captura conectado ao equipo. É o encargado de recoller os datos de calibración desde os dous dispositivos e de calcular, a partires deles, a transformación afín que é necesario aplicarlle aos datos de captura dun dos dispositivos para que traballe no mesmo sistema de coordenadas que o outro. O diagrama de clases que representa esta responsabilidade móstrase na figura 5.5.

A implementación deste módulo ten unha compoñente experimental considerable. Para un mellor entendemento desta parte, abórdase máis en profundidade no capítulo seguinte, *Deseño Experimental*.

Responsabilidade: Recollida de datos - *Data Gathering*

O módulo de *Recollida de datos* é o encargado de adquirir os datos de captura directamente do(s) dispositivo(s). O diagrama de clases que representa esta responsabilidade móstrase na figura 5.6.

No contorno de traballo cun só dispositivo este módulo establece e xestiona a conexión co mesmo, e recolle os datos que este lle envía durante a gravación dunha sesión de interpretación. No contorno de dous dispositivos de captura, o módulo recolle os datos dos mesmos, e aplica os parámetros de transformación para que traballen no mesmo sistema de coordenadas. Unha vez transformados, fusiona os datos dos dous dispositivos para que a saída sexa idéntica á do contorno cun só dispositivo. Desta maneira, que no equipo haxa conectados un ou dous dispositivos só é relevante para este módulo e para o de calibración, independizando ao resto do subsistema deste feito.

En calquera dos dous contornos, outra das funcións que leva a cabo este módulo é a de mostrar ao usuario os datos de captura de maneira visual. Para iso utilizouse a característica da tecnoloxía escollida, *WPF*, que permite mostrar unha escena tridimensional. Así, representando cada articulación do esqueleto cunha esfera e cada óso cun cilindro, e situándoos na escena tridimensional segundo os datos de captura que se van recollendo, é posible visualizar en tempo real e de maneira moi fiel que está visualizando o sistema durante a sesión de interpretación.

Os datos que utiliza para esta mostra visual son os mesmos que lle envía ao resto de módulos, asegurando así que os datos que se comparten son coherentes co que o usuario está vendo.

Responsabilidade: Recoñecemento de voz - *Speech Recognition*

O módulo de *Recoñecemento de voz* é o encargado de capturar ordes de voz desde o dispositivo e transformalas en accións dentro do subsistema Servidor. O diagrama de clases que representa esta responsabilidade móstrase na figura 5.7.

A libraría de *Microsoft* para interacción con *Kinect* proporciona unha serie de instrumentos moi sinxelos de utilizar para levar a cabo esta tarefa. Cabe salientar o feito de que foi necesario instalar un módulo especial para recoñecemento da lingua española, polo que as ordes de voz poden ser neste idioma ou en lingua inglesa. A elección dos comandos que se incluíron neste módulo pretendía mellorar a usabilidade da aplicación. Así, no caso de que usuario e intérprete sexan a mesma persoa, é dicir, que a aplicación sexa utilizada por unha soa persoa, esta pode iniciar a aplicación e controlar practicamente todo desde a zona de captura sen ter que achegarse ao ordenador cada vez que necesita executar un novo comando.

Responsabilidade: Coordinación do subsistema Servidor

A tarefa de coordinar ao resto de módulos, de manter o estado a aplicación e de facilitar a comunicación entre presentadores recae neste último módulo, cuxo diagrama de clases se mostra na figura 5.8.

Este módulo é certamente diferente ao resto de expostos anteriormente. Está intimamente ligado coa clase principal do subsistema Servidor, *ServerMain*. Esta clase é a que inicia a aplicación e tamén é a encargada de manter o seu estado. Nun primeiro momento a responsabilidade de manter o estado da aplicación recaía no *ServerPresenter* pero durante o desenvolvemento chegouse á conclusión de que este último só debía asumir tarefas de Presentación. Para transferir esta responsabilidade de *ServerPresenter* a *ServerMain* utilizouse un patrón **Proxy** de maneira que toda a lóxica referida ao estado da aplicación pasa á clase *ServerMain*. En *ServerPresenter* só quedan as referencias aos métodos e variables relacionados co estado da aplicación, delegando a responsabilidade en *ServerMain*. Por último e para completar a implementación formal do patrón *Proxy* foi necesario crear unha interface, *IServerMain*, que contén a colección de métodos e propiedades relacionados coa responsabilidade de manter o estado da aplicación, e que deben implementar tanto *ServerPresenter* como *ServerMain*. A partires deste momento, o resto de Presentadores comunícanse coa aplicación a través desta interface diminuíndo o acoplamento entre este módulo e o resto que compoñen o subsistema Servidor.

5.2.3. Diagrama de secuencia

Na figura 5.9 preséntase o diagrama de secuencia que representa os casos de uso CU.01 e CU.02. Este diagrama pretende dar unha visión dinámica de como funciona o subsistema Servidor ao longo do tempo, tendo en conta as interaccións do usuario.

Decidiuse mostrar os dous casos de uso no mesmo diagrama porque o CU.02 estende ao CU.01, de maneira que o CU.02 só ten sentido se antes tivo lugar o CU.01.

Para facer máis sinxelo e entendible o diagrama eliminároнse as Vistas, tomando a cada Presentador como representante da responsabilidade que ostenta. De todas maneiras, traballando cunha arquitectura MVP as Vistas só teñen comunicación co seu propio Presentador, polo que non aportarían información relevante ao diagrama.

5.2.4. Interface gráfica

O deseño da interface gráfica do Servidor xurdiu en base á arquitectura escollida. O feito de ter dividido o subsistema en responsabilidades, cada un cunha Vista independente permitiu deseñar unha interface gráfica na que todo o necesario está en pantalla e dispoñible para o usuario en todo momento. Na figura 5.10 móstrase un instante de uso, mentras se está a gravar unha sesión de interpretación. Neste imaxe pódese observar como a Vista *ServerView* representa á ventá da aplicación, e o resto de Vistas están organizadas no seu interior de maneira que todas son accesibles e visibles para o usuario en todo momento, mellorando a usabilidade. Na figura 5.11 móstrase a aplicación no momento de calibrar o sistema. Unha vez este rematou a calibración a Vista de *DataGathering* toma o lugar da vista de *DeviceCalibration*.

5.3. Cliente

O subsistema Cliente será o encargado de transformar os datos de captura en datos de animación no contorno dun software de xeración de gráficos por ordenador, concretamente *Autodesk Maya* e *Blender*.

Tal como se fixo co subsistema Servidor, en primeiro lugar presentarase o patrón de arquitectura que indicará como se van organizar as responsabilidades no subsistema Servidor. Para continuar mostrarase a organización en clases, para rematar con dous diagramas de secuencia que mostran como se comporta a aplicación na interacción co usuario.

5.3.1. Patrón de arquitectura

Para o patrón de arquitectura do Cliente consideráronse os mesmos que para o subsistema Servidor, *MVC* e *MVP*.

Para abordar a elección do patrón de arquitectura convén aclarar en primeiro lugar algúns aspectos peculiares do subsistema Cliente, por formar parte dun software de xeración de gráficos por ordenador: existe unha entidade que xa se explicou anteriormente, coñecida nos software de xeración de gráficos por ordenador como *Skeleton*, que vai estar composta por ósos. Nestes ósos é onde se van transferir ao longo do tempo os datos de animación. Tendo isto en conta, sería completamente innecesario considerar a creación de clases que conteñan esta información, polo que se tomou a decisión de considerar estas entidades como clases que forman par-

te do Modelo. Esta concepción orixinal do modelo de clases adaptado ao entorno dos software de xeración de gráficos por ordenador elimina complexidade no código, mantendo a mesma funcionalidade. Ademais, estas entidades van ser actualizadas fotograma a fotograma e isto vai ter a súa correspondencia no que o usuario ve, e polo tanto na Vista.

Tendo isto en conta, o patrón de arquitectura máis adecuado para este caso é *MVC*, polo contacto entre o Modelo e a Vista.

5.3.2. Deseño e implementación

Nesta sección abórdase a organización en clases do subsistema Cliente dentro da arquitectura *MVC*. Tal como se fixo no subsistema Servidor continúase utilizando un código de cores para distinguir en que parte da arquitectura se encadra cada clase.

O diagrama de clases da figura 5.12 mostra unha visión global da organización das clases no deseño proposto para o subsistema Cliente.

Para este subsistema extráense as seguintes responsabilidades:

- Creación de esqueletos/*Skeleton Creator*
- Creación de animación/*Animation Creation*
- Adquisición de datos/*Data Adquisition*
- Coordinación do subsistema Cliente

A continuación detállanse as diferentes responsabilidades por separado, e tal como se fixo co subsistema Servidor, comentaranse aqueles detalles de implementación que se consideren relevantes.

Responsabilidade: Adquisición de datos - *Data Adquisition*

O módulo de Adquisición de datos ten asignadas dúas tarefas: en primeiro lugar, é o encargado de establecer e xestionar a conexión coa parte Servidor para recoller os datos de captura que este pon á súa disposición, e en segundo lugar encárgase de ler o ficheiro con datos de captura. En calquera dos dous casos os datos adquiridos son transferidos ao módulo de creación de animación para que este realice a súa función.

A lóxica desta responsabilidade está centralizada no controlador: *DataAdquisition*.

Responsabilidade: Creación de esqueletos - *Skeleton Creator*

O módulo de Creación de esqueletos é o encargado de crear o *Skeleton* composto por aqueles ósos cos que traballa o *SDK* de *Microsoft* para aloxar a animación que se recibe desde o subsistema Servidor ou desde un ficheiro.

A lóxica desta responsabilidade está centralizada no controlador: *Skeleton-Creator*.

Responsabilidade: Creación de animación - *Animation Creation*

O módulo de Creación de Animación encárgase de tomar os datos de captura recollidos polo módulo de adquisición de datos, transformalos en datos de animación e transferilos ao *Skeleton*, óso por óso.

Con respecto á implementación, é necesario salientar que tivo aproximacións diferentes en *Autodesk Maya* e en *Blender*, debido en gran parte a como está concibida a creación de animación en cada un dos software citados.

Por unha banda, *Maya* non traballa directamente con *quaternions* senón con ángulos de *Euler*, polo que foi necesario utilizar unha función do *SDK* desta aplicación para transformar as rotacións recibidas en *quaternions* a ángulos de *Euler*.

Por outra, en *Blender* si é posible traballar directamente con *quaternions* polo que neste caso non foi necesaria a transformación dos datos recibidos.

Ademais, *Maya* permite asignar datos de posición aos ósos, polo que é posible xerar os dous tipos de animación contemplados nos requisitos. *Blender*, polo contrario, só acepta datos de rotación para os seus ósos, polo que para este software só foi posible levar a cabo a xeración de animacións baseadas en rotacións.

Por último, é necesario aclarar que non foi posible desenvolver parte dos requisitos opcionais para *Blender* por superar o tempo asignado ao proxecto. Estes requisitos si foron implementados na versión do cliente para *Maya*.

A lóxica desta responsabilidade está centralizada no controlador: *Animation-Creator*.

Coordinación do subsistema Cliente

Por último, a responsabilidade de coordinar a aplicación recae sobre *Client-Controller*. Esta clase é a encargada de iniciar a aplicación Cliente e de manter o estado desta mentres está en execución. Ademais, no deseño de clases do subsistema Cliente, *ClientController* adquire tamén a responsabilidade de **Fachada** entre a Vista e o resto de Controladores.

5.3.3. Diagrama de secuencia

Nos diagramas de secuencia da figura 5.13 e da figura 5.14 móstranse de maneira dinámica a interacción do usuario co subsistema Cliente nos casos de uso CU.03 e CU.04 respectivamente.

5.3.4. Interface gráfica

Para a implementación da Vista no subsistema Cliente foi necesario adaptarse ás particularidades dos software de xeración de gráficos por ordenador. Así, tal como se pode ver na figura 5.15 a Vista en *Autodesk Maya* é unha ventá separada na que se pode observar o acceso ás diferentes funcionalidades implementadas. Por outra banda, *Blender* non foi posible realizar a implementación nunha ventá e foi necesario incluila como parte da propia interface de usuario do software, tal como se observa na figura 5.16.

5.4. Comunicación Cliente-Servidor

Para a comunicación entre subsistema Cliente e subsistema Servidor tomouse a decisión de utilizar o paradigma **Produtor-Consumidor**. Este paradigma adáptase perfectamente a este caso, xa que é necesario asegurar que todos e cada un dos datos capturados polo subsistema Servidor chegan ao subsistema Cliente.

O funcionamento do paradigma Produtor-Consumidor tal como se desenvolveu para este sistema é o seguinte:

O subsistema Servidor, que ten o rol de produtor, recolle datos de captura e vainos almacenando nun *buffer* a medida que os adquire. Neste caso o citado *buffer* implémentase como unha cola. Esta cola ten un límite de datos que pode aloxar, polo que no caso de que se chegue a ese límite comezaranse a eliminar os primeiros datos que entraron na mesma. En todo caso, esa perda de datos é asumible xa que existe a posibilidade de almacenar todos os datos de captura nun ficheiro, en caso de que non sexa necesario establecer a conexión co subsistema Cliente.

Por outra banda, o subsistema Cliente accede ao *buffer* para adquirir datos de captura. Mentres o *buffer* conteña datos o subsistema Cliente recolleraos e fará uso deles para xerar a animación. Se o *buffer* se baleira, quédase á espera ata o momento no que volve a haber datos, continuando entón coa adquisición dos mesmos.

Este paradigma explicase de maneira dinámica no diagrama de secuencia da

figura 5.17. É necesario destacar que neste diagrama non se mostran clases senón partes da arquitectura xeral do sistema para un mellor entendemento do paradigma.

Na figura 5.18 e na figura 5.19 pódese observar a comunicación do subsistema Servidor co Cliente no contorno de *Autodesk Maya* e de *Blender*, respectivamente.

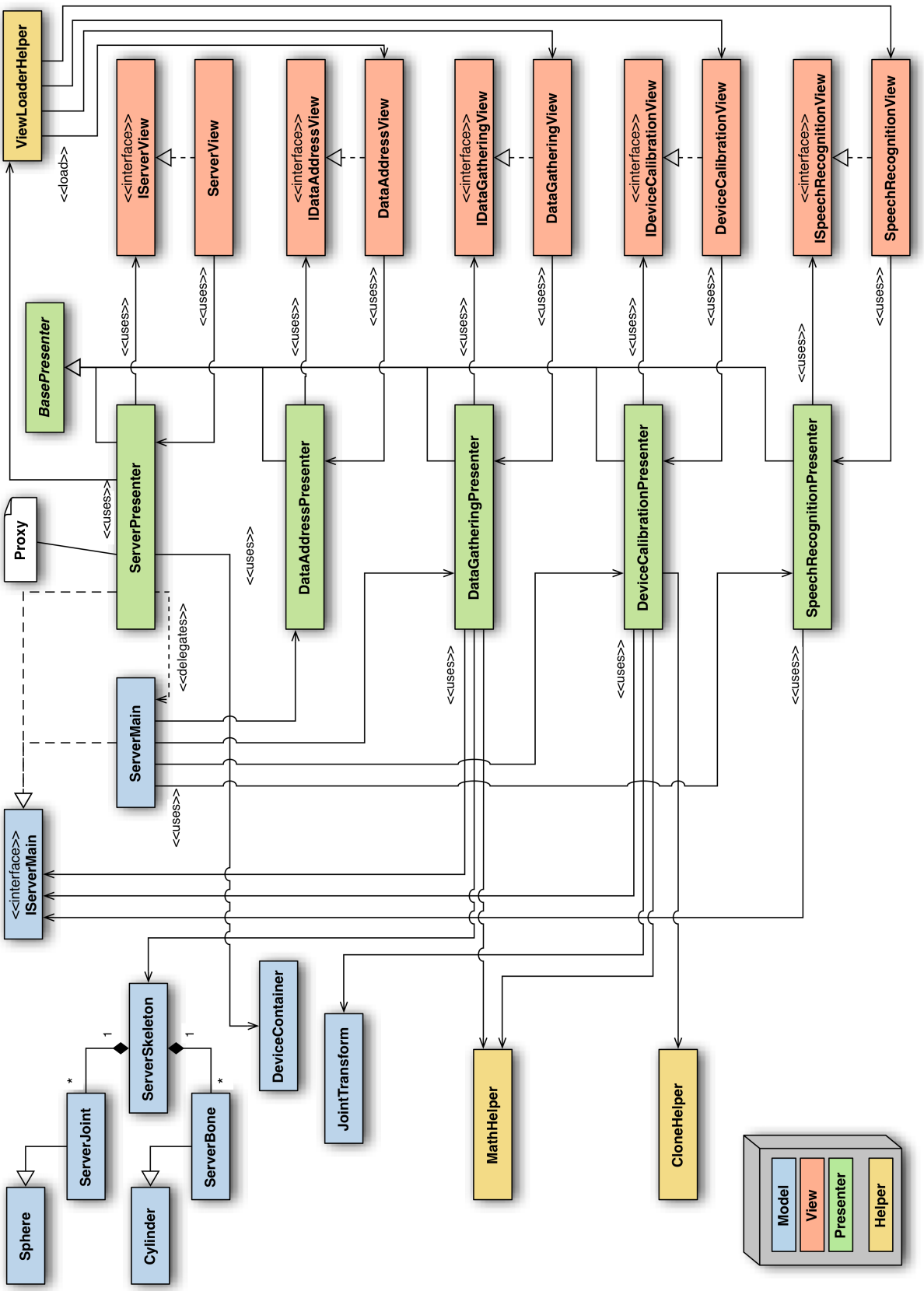

Figura 5.3: Diagrama de clases xeral do subsistema Servidor

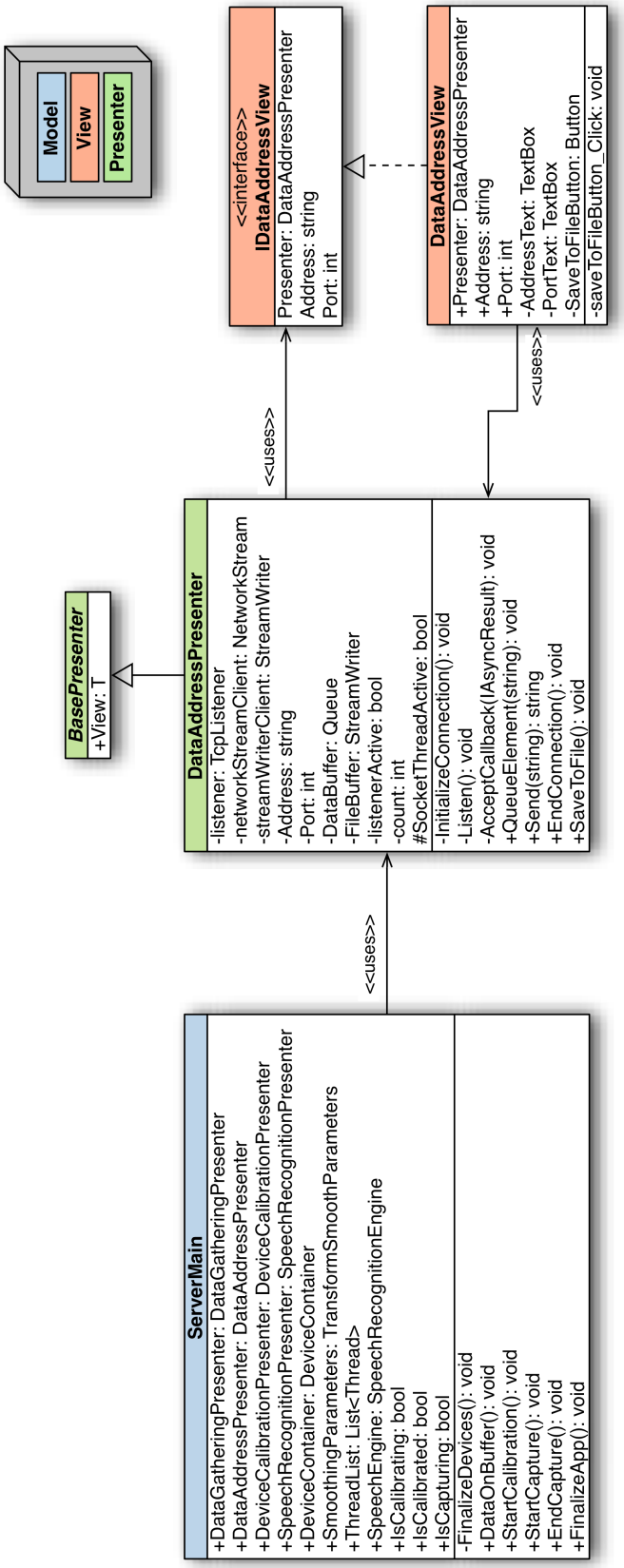

Figura 5.4: Diagrama de clases que representa a responsabilidade **Xestión de datos**

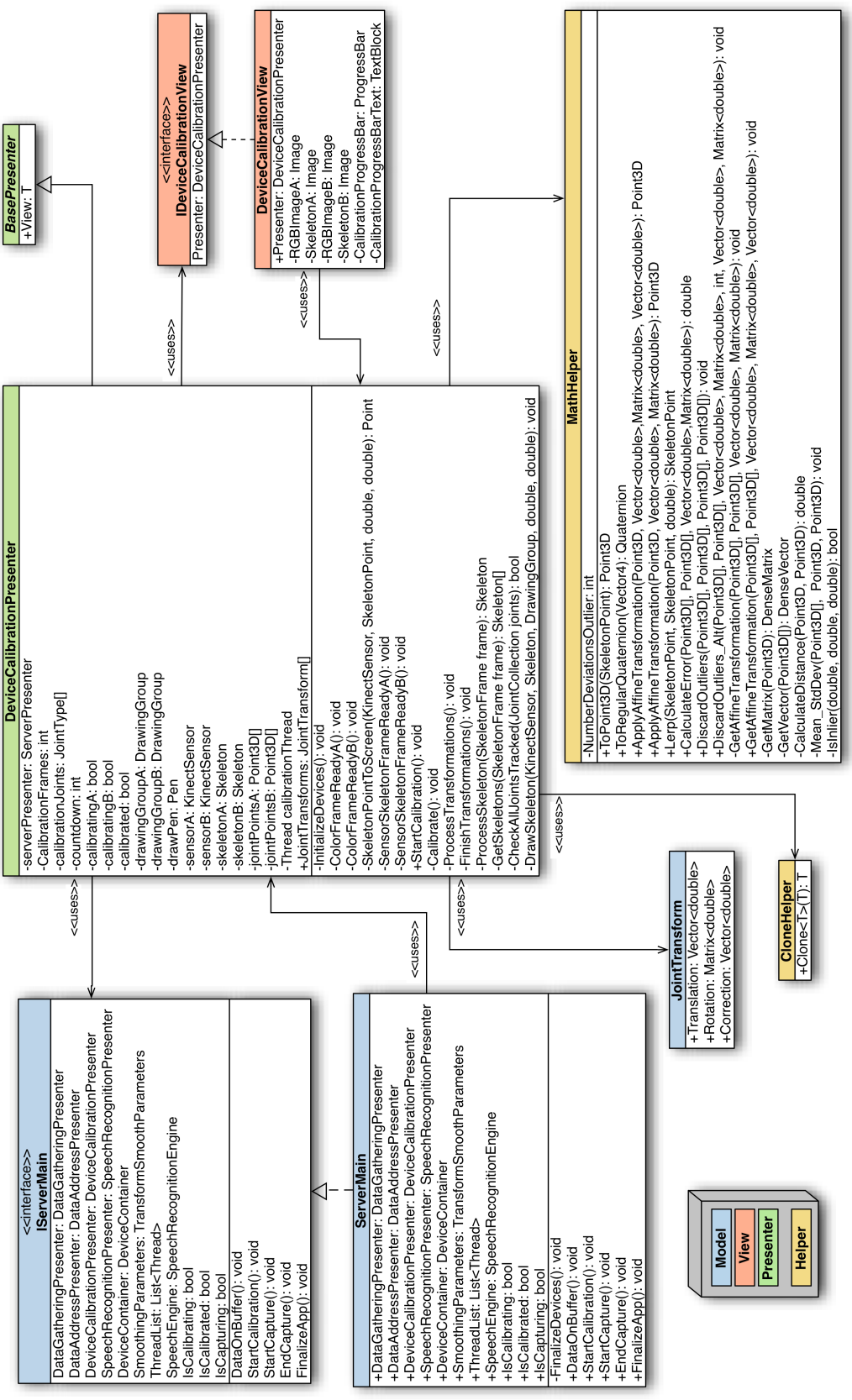

Figura 5.5: Diagrama de clases que representa a responsabilidade **Calibración de dispositivos**

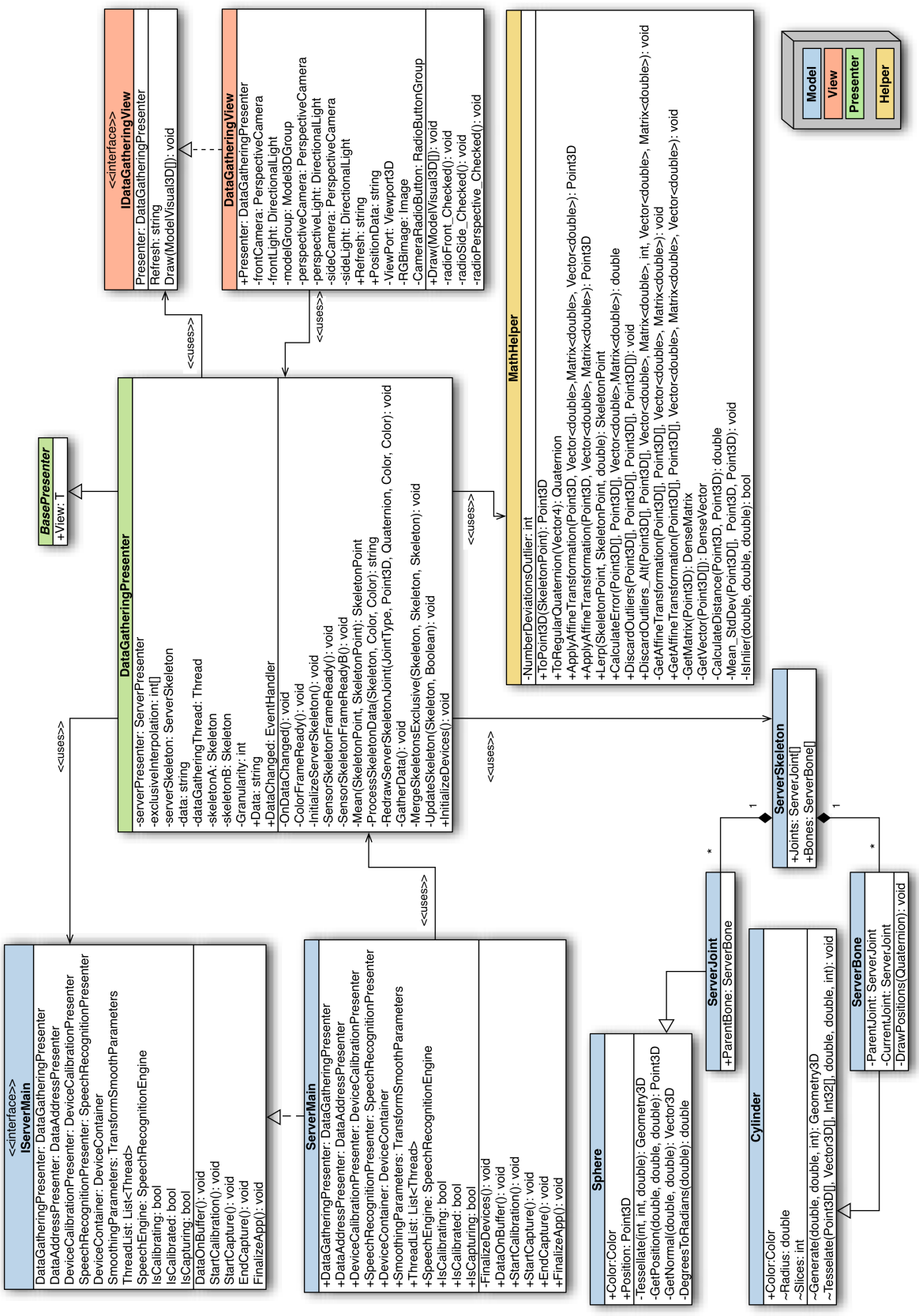

Figura 5.6: Diagrama de clases que representa a responsabilidade **Recollida de datos**

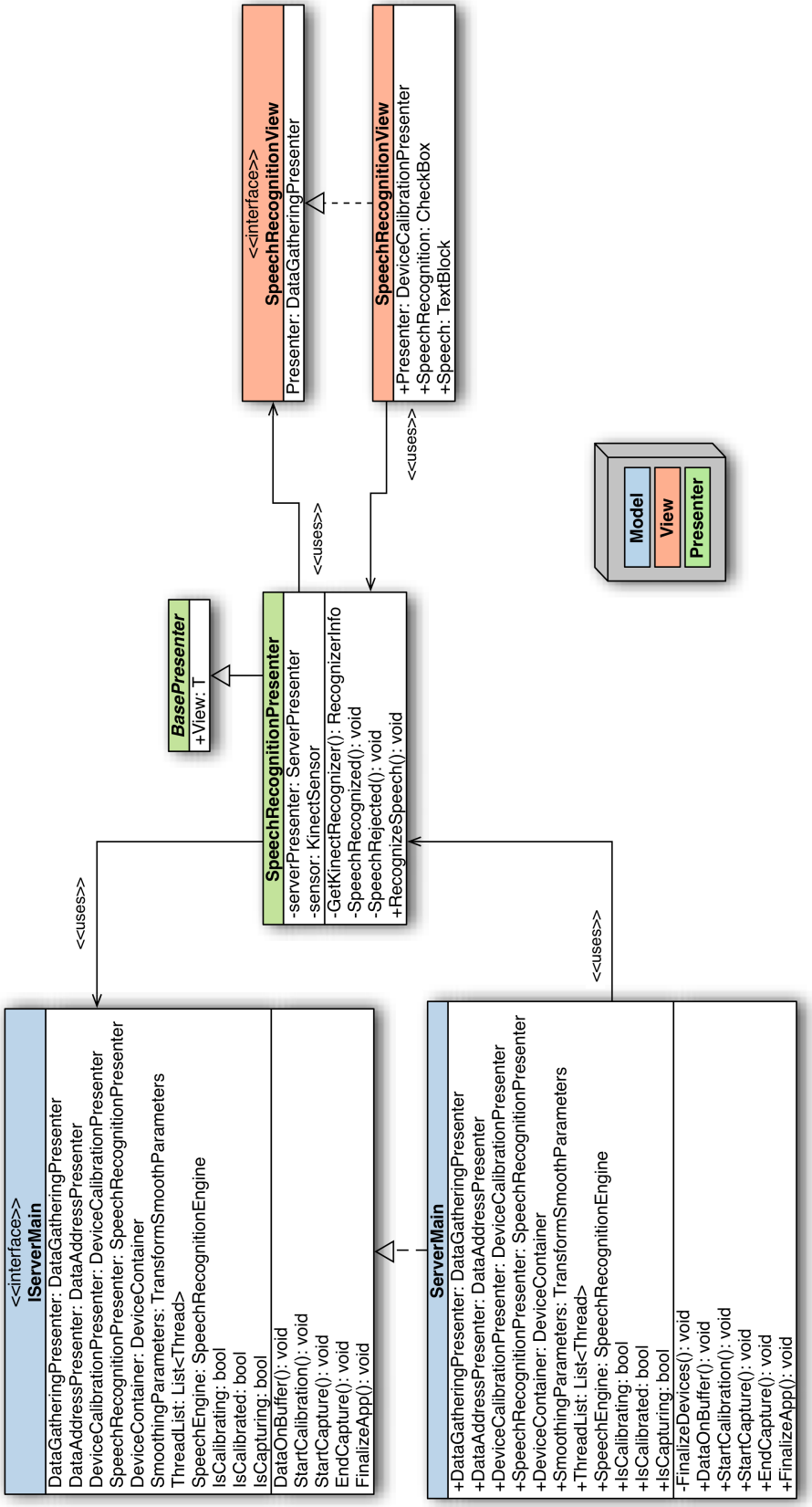

Figura 5.7: Diagrama de clases que representa a responsabilidade **Recoñecemento de voz**

Figura 5.8: Diagrama de clases que representa a responsabilidade **Coordinación do subsistema Servidor**

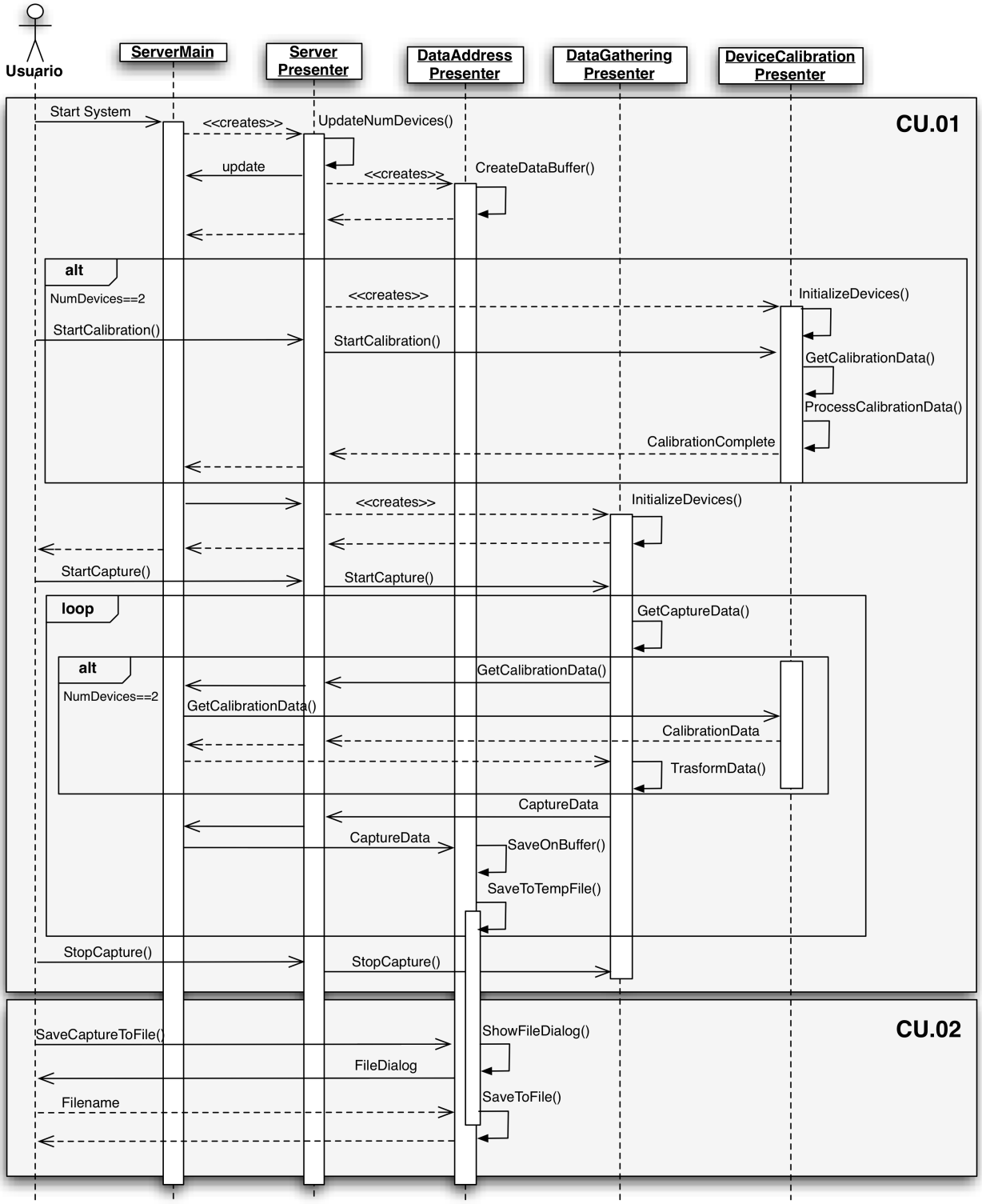

Figura 5.9: Diagrama de secuencia que representa o CU.01 e o CU.02

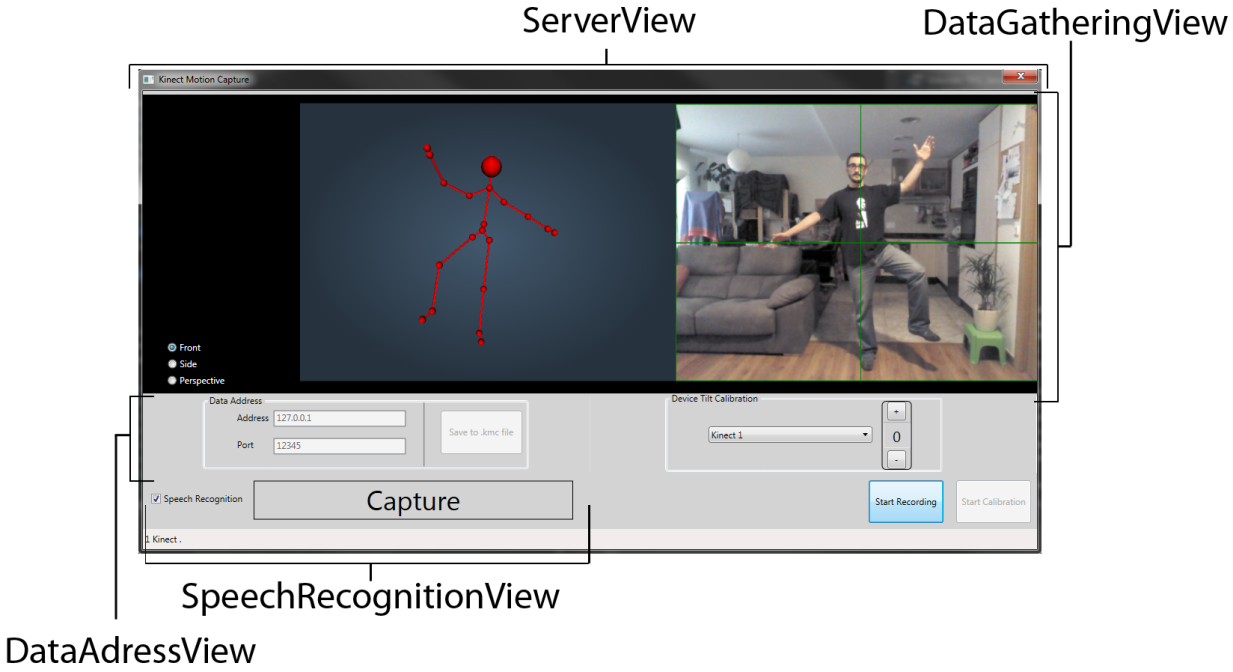

Figura 5.10: Vista xeral da interface gráfica da aplicación durante a gravación dunha sesión de interpretación

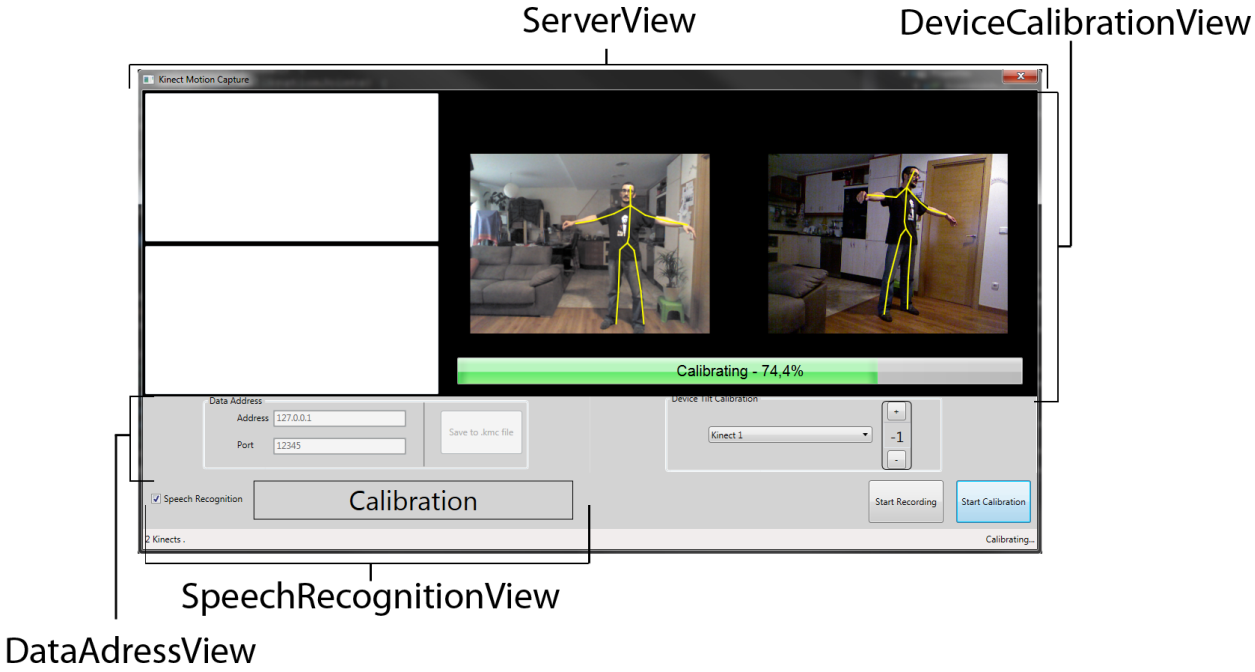

Figura 5.11: Vista xeral da interface gráfica da aplicación durante a calibración do sistema

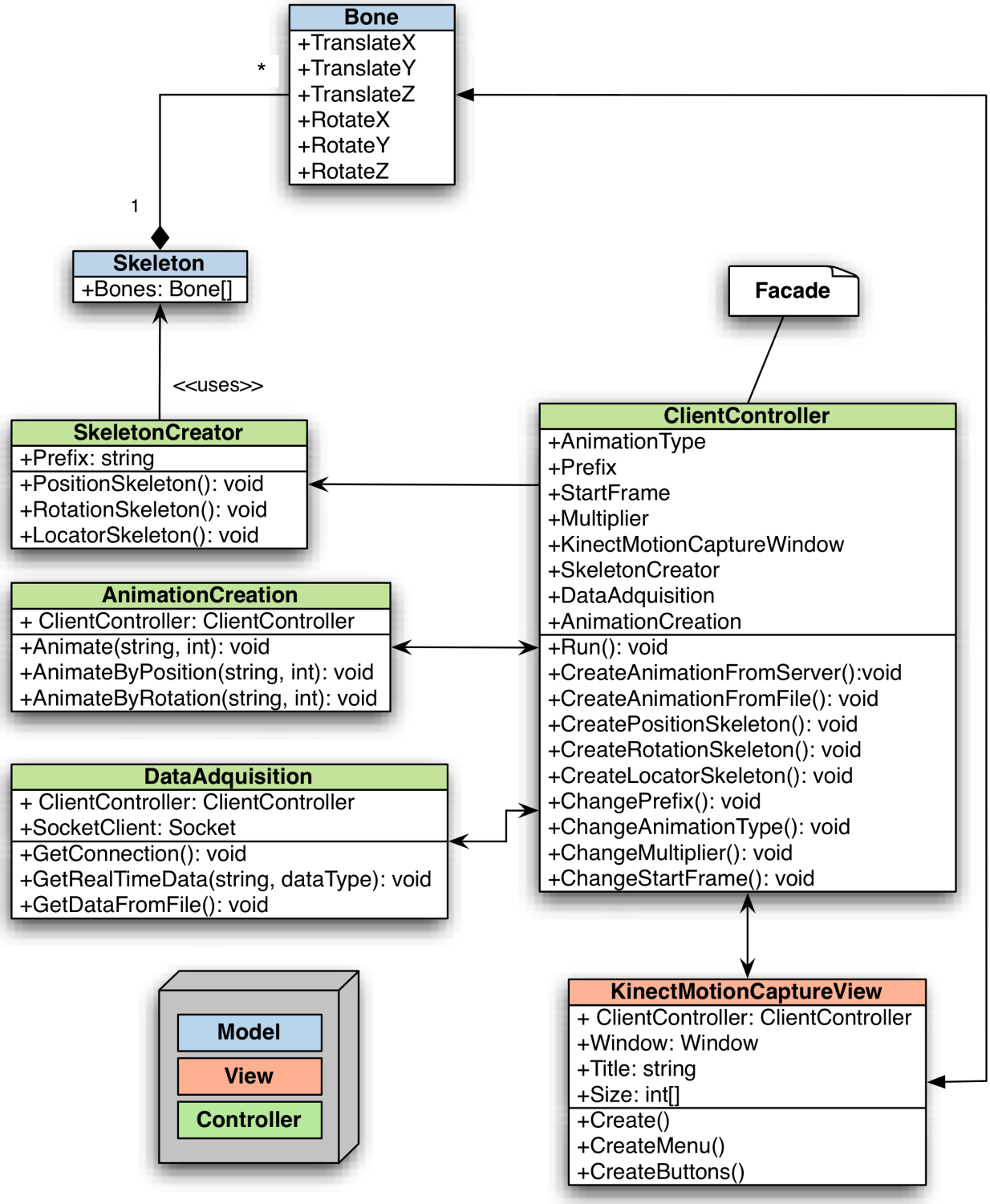

Figura 5.12: Diagrama de clases do subsistema Cliente

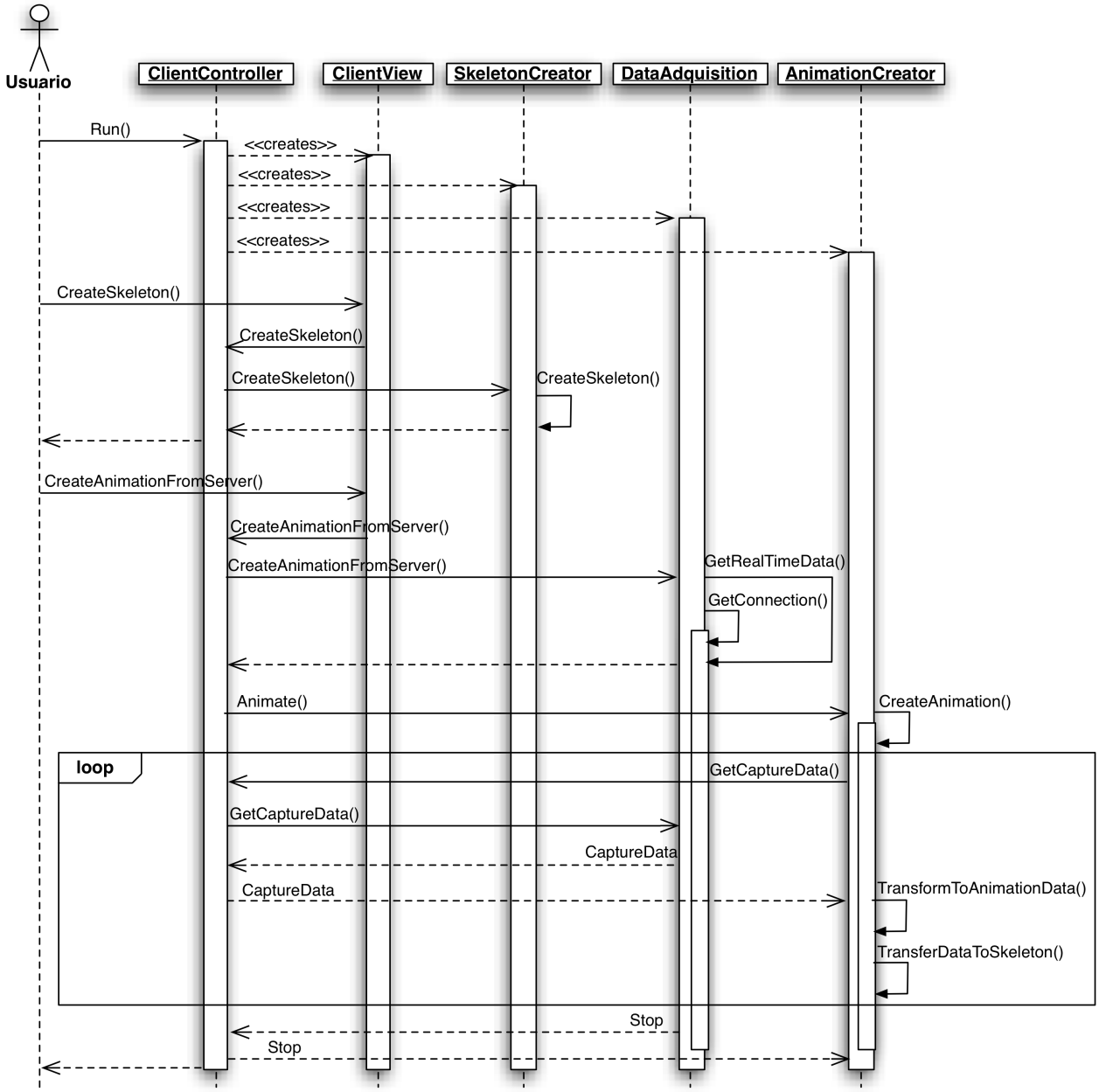

Figura 5.13: Diagrama de secuencia que representa o CU.03

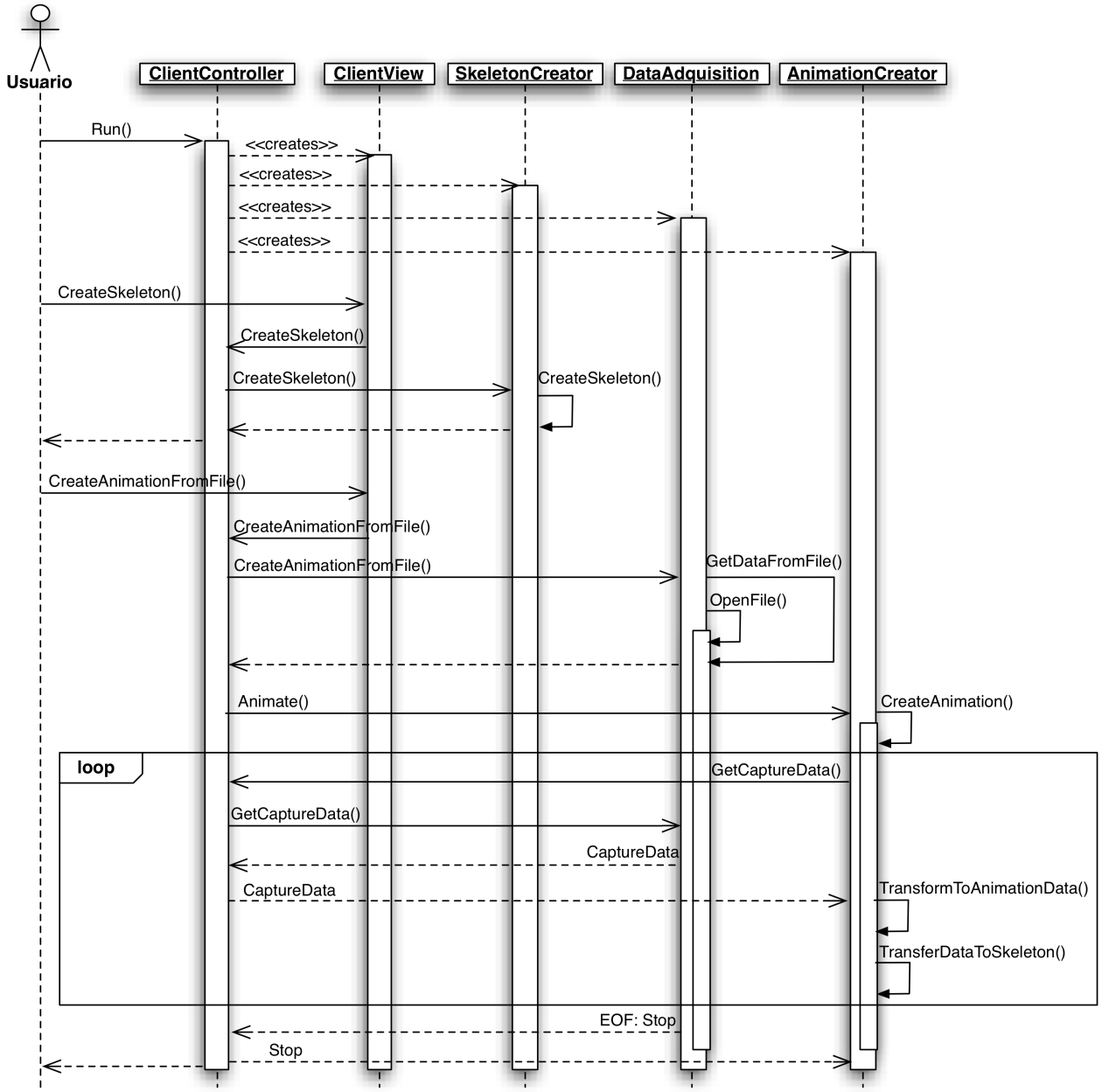

Figura 5.14: Diagrama de secuencia que representa o CU.04

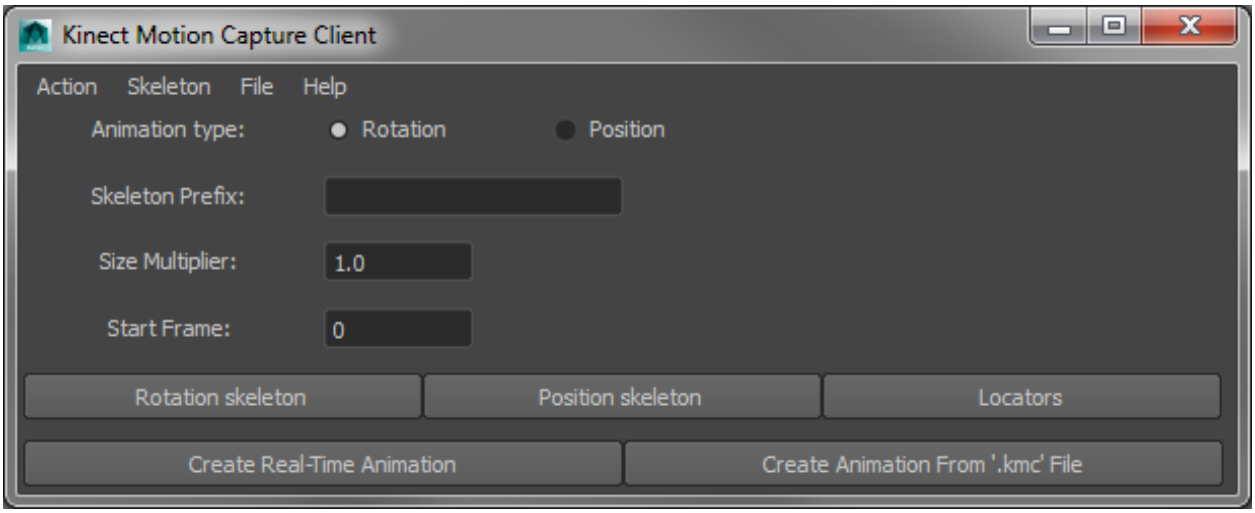

Figura 5.15: Vista do subsistema Cliente en *Autodesk Maya*

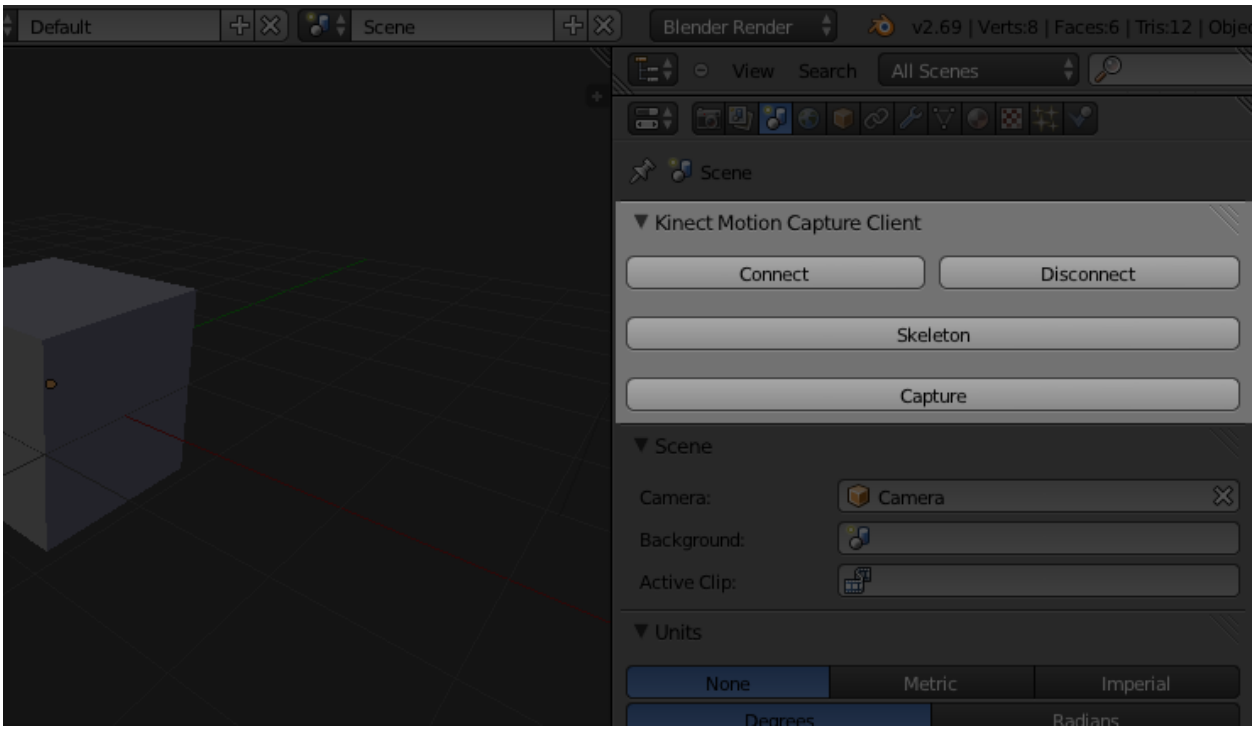

Figura 5.16: Vista do subsistema Cliente en *Blender*

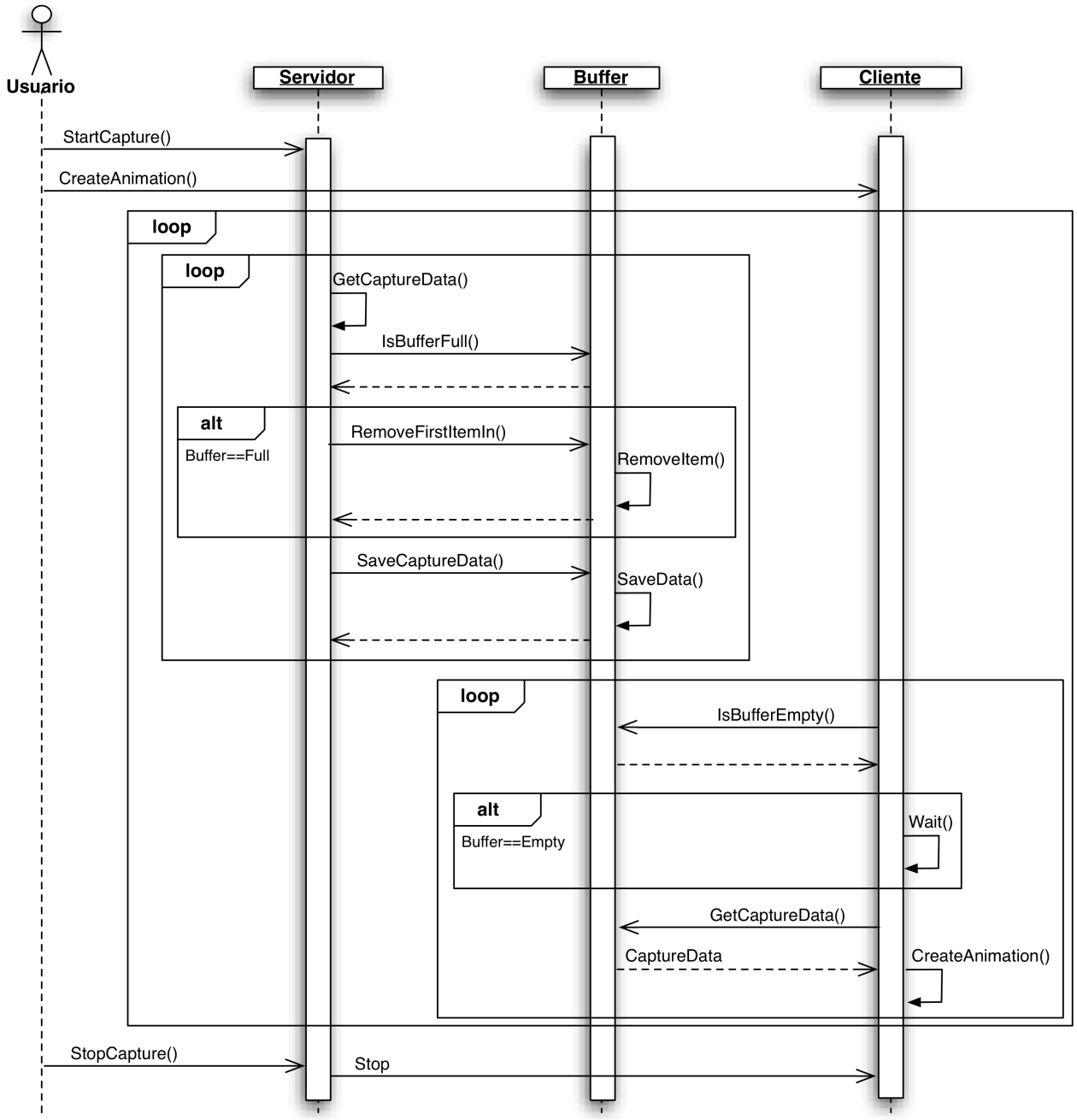

Figura 5.17: Diagrama de secuencia que representa a comunicación entre Cliente e Servidor

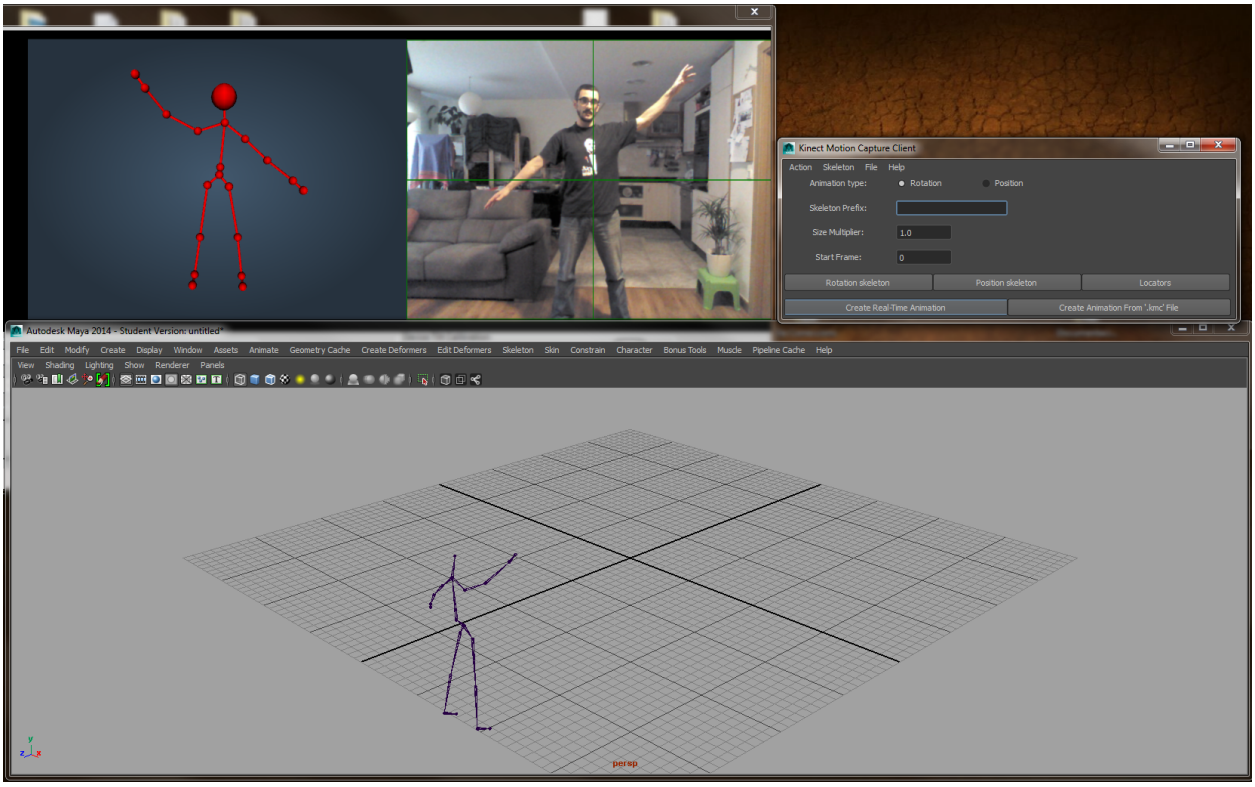

Figura 5.18: Cliente en *Autodesk Maya* e Servidor durante a gravación dunha sesión de interpretación

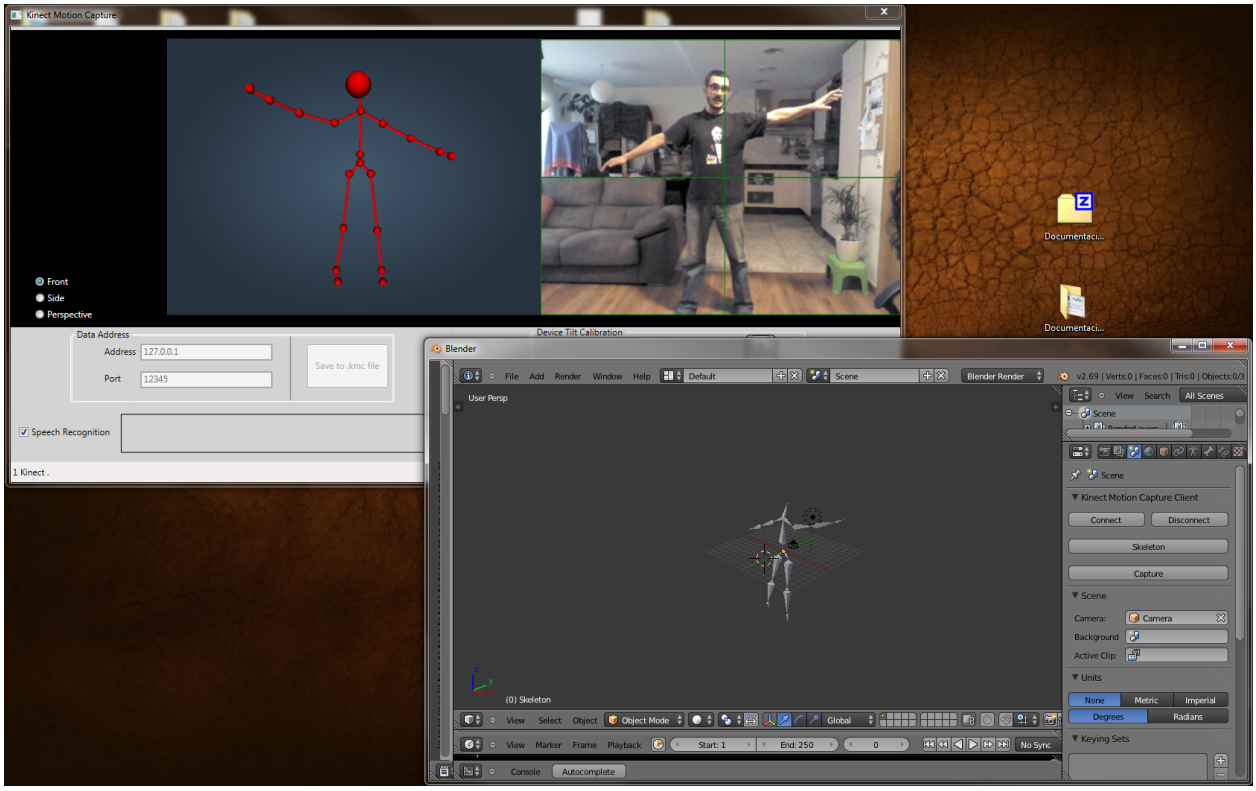

Figura 5.19: Cliente en *Blender* e Servidor durante a gravación dunha sesión de interpretación

Capítulo 6

Deseño Experimental

Neste capítulo teñen cabida todos aqueles aspectos que supuxeron un desafío técnico para a consecución do proxecto, así como as decisións que se tomaron para resolvelos. Para comezar expóñense as razóns e os beneficios de ter desenvolvido un prototipo vertical. Continuarase explicando os problemas técnicos que se atoparon á hora de abordar as animacións baseadas en rotacións. Por último, preséntanse os problemas que xurdiron para realizar a calibración no contorno con dous dispositivos de captura. A continuación proporcionase unha explicación máis detallada destas partes do proxecto.

6.1. Prototipo vertical

A razón de realizar un prototipo vertical [37] é avaliar a viabilidade daquelas funcións dun produto software que se van abordar con maior incerteza. Para que un prototipo vertical sexa xustificable deben probarse só funcións relevantes para a consecución do proxecto.

Nos inicios deste proxecto, existían certos aspectos fundamentais que o equipo de desenvolvemento non podía asegurar que fosen viables en tempo e forma. Propúxose entón a realización dun prototipo vertical para avaliar os seguintes puntos do desenvolvemento:

- Adquisición de datos de posición desde un dispositivo utilizando a libraría de interacción con *Kinect* de *Microsoft* na tecnoloxía *.NET*.
- Adquisición de datos de posición desde dous dispositivos utilizando a libraría de interacción con *Kinect* de *Microsoft* na tecnoloxía *.NET*.
- Conexión desde unha aplicación en *.NET* a unha aplicación en *Python* no contorno dun software de xeración de gráficos por ordenador, concretamente *Autodesk Maya*.
- Envío dos datos adquiridos desde *Kinect* a *Autodesk Maya* e velocidade de transferencia de datos por segundo.
- Transferencia destes datos a un obxecto dentro de *Maya* ao ritmo de recepción dos mesmos.

A partires da formación adquirida para realizar o prototipo e da consecución deste extraéronse as seguintes conclusións técnicas:

- A adquisición e envío de datos desde un *Kinect* a *Autodesk Maya* é de 32 fotogramas de captura por segundo. O fluxo de fotogramas nunha película de cine convencional é de 24 fotogramas por segundo [41], polo que o fluxo acadado é máis que suficiente para realizar unha animación a partires dos datos de captura.

- A adquisición e envío de datos desde dous dispositivos é posible e proporciona o mesmo fluxo de datos que no caso dun só dispositivo.

- Non é posible a conexión de dous dispositivos *Kinect* nun equipo se non conta con dúas controladoras USB. Incluso equipos con catro e seis portos USB poden non permitilo por formar todos parte da mesma controladora USB.

A realización do prototipo vertical probou que unha parte importante das funcionalidades que require o proxecto son posibles, e polo tanto asegura a súa viabilidade. Ademais, o feito de realizar o prototipo axudou ao equipo de desenvolvemento a ter unha visión máis global do proxecto e a realizar un deseño máis adecuado para o problema proposto.

6.2. Animación baseada en rotacións

O *SDK* de *Microsoft* para *Kinect* permite traballar con rotacións absolutas ou xerárquicas: nas **absolutas** a rotación de cada óso é independente do que está por enriba na xerarquía e, nas **xerárquicas** o valor de rotación é con respecto ao óso que está por enriba na xerarquía.

Por outra banda, no software de xeración de gráficos por ordenador o esqueleto é xerárquico, de maneira que cando se lle aplica unha rotación a un óso, todos os que están por debaixo na xerarquía rotan con el.

Tendo isto en conta, para obter unha animación coherente coa captura no software de xeración de gráficos por ordenador é necesario utilizar os datos de rotación xerárquicos.

Durante o desenvolvemento, cando se comezou a traballar sobre a animación baseada en rotacións atopouse que, ao transferir os datos de rotación ao esqueleto, tanto en *Autodesk Maya* como en *Blender*, as diferentes partes do mesmo distribuíanse de maneira incorrecta (Fig. 6.1).

O problema reside en que o *SDK* de *Microsoft* considera que os ósos do esqueleto parten dun valor de rotación igual a cero, e para representar isto en *Autodesk*

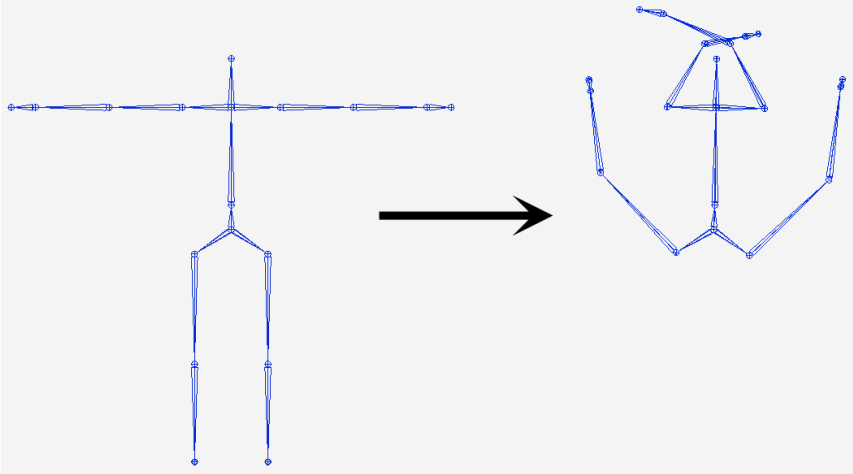

Figura 6.1: Esqueleto con problema de rotacións en *Autodesk Maya* despois de transferir os datos de captura.

Maya é necesario crear o esqueleto de maneira que todos os osos teñan a orientación do eixo Y, é dicir, cara arriba. Este feito provoca que ao crear o esqueleto no software de xeración de gráficos por ordenador este teña a forma que se mostra na figura 6.2.

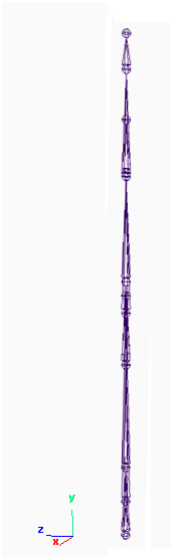

Figura 6.2: Esqueleto cos ósos no eixo Y que soluciona parte do problema das rotacións en *Autodesk Maya*.

Co problema das rotacións solventado xorde outro obstáculo, neste caso para o animador que deba aplicar esta rotación a un modelo poligonal. A práctica habitual para crear modelos poligonais de personaxes para animación é construílos en

posición erguida cos brazos abertos en perpendicular co corpo tal como se mostra na parte esquerda da figura 6.1. O feito de facelo así é que é a posición na que as partes do corpo da personaxe están o suficientemente lonxe unhas de outras, de maneira que se poida asociar cada óso do esqueleto a cada unha desas partes sen influenciar ás outras. Se partimos do esqueleto que nos proporciona a solución para o problema das rotacións, a tarefa de asocialo cun modelo poligonal tórnase moito máis complexa. Por esta razón, foi necesario idear unha maneira que permitira resolver os dous problemas a un tempo.

A solución final para poder xerar animacións baseadas en rotacións coherentes coa captura foi crear un esqueleto cos ósos en vertical, e aplicarlle os datos de animación procedentes dunha captura para que o esqueleto tomara a forma adecuada (Fig. 6.3). Co esqueleto creado, anotáronse os valores de rotación de todos os ósos para introducilos na creación do esqueleto para animacións baseadas en rotacións. Desta maneira, cando se crea ese esqueleto, xa ten de maneira nativa as rotacións base coas que traballa *Kinect* e polo tanto pódese asociar a calquera modelo creado coa forma descrita sen que este se deforme sen control.

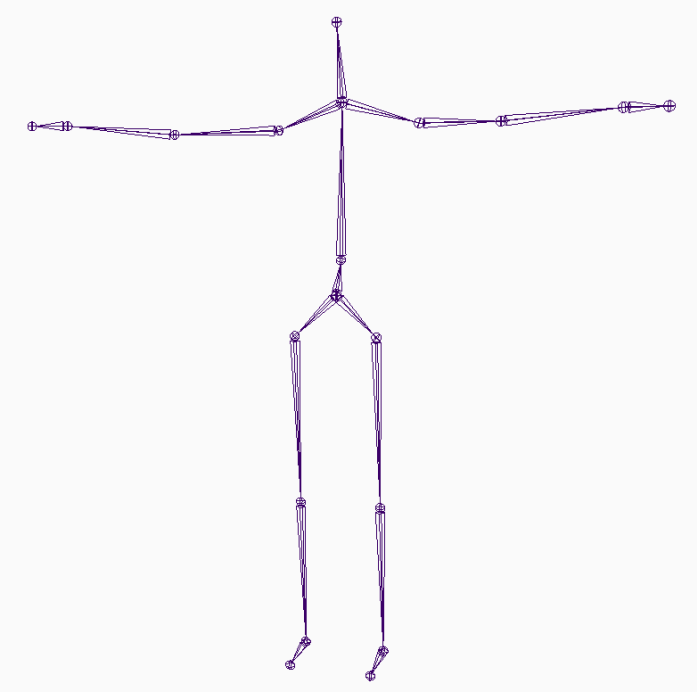

Figura 6.3: Esqueleto cos ósos no eixo Y que soluciona parte do problema das rotacións en *Autodesk Maya*.

6.3. Calibración

Un dos obxectivos do proxecto é que o sistema poida realizar captura de datos con alomenos dous dispositivos. Se se realiza unha sesión de interpretación gravando a un intérprete utilizando dous dispositivos á vez, obteranse dous conxuntos de datos de captura referidos aos mesmos puntos no espazo, correspondentes coas articulacións do intérprete ao longo do tempo. Como cada un dos dispositivos traballa nun sistema de coordenadas diferente cuxo centro é o propio dispositivo, é necesario transformar os puntos capturados para situalos no mesmo sistema de coordenadas, como se fosen obtidos desde un só dispositivo. A continuación preséntase tanto a etapa de recollida de datos como a do cálculo da transformación. Estas dúas etapas conforman o que denominaremos **calibración do sistema**.

6.3.1. Captura de datos de calibración

Para conseguir que os dous dispositivos traballen no mesmo sistema de coordenadas é necesario ter o mesmo conxunto de puntos capturado desde os dous dispositivos a un tempo. Como este é un sistema sen marcas vai existir un certo erro na identificación da mesma articulación por parte de dispositivos diferentes. Ademais, pode existir un erro engadido debido á leve deformación da imaxe, producida por estar visualizando ao mesmo suxeito desde ángulos diferentes.

Ao non traballar cun dispositivo de precisión asumimos que o feito de que cada *Kinect* recoñeza dunha maneira diferente as posicións das articulacións no mesmo suxeito, é consecuencia das características propias do hardware, como particularidades no emisor ou no receptor infravermello. Tamén poden vir por diferencias no entorno, xa que ao estar os dispositivos situados en ángulos diferentes, o fondo da escena é diferente, podendo influír en como é captado o corpo do intérprete. Tendo isto en conta tomouse a decisión de recoller os puntos para calibrar directamente dunha captura co intérprete. Desta maneira no cálculo da transformación vai incluído o erro debido ao dispositivo, minimizándoo de maneira considerable.

Por outra banda, tomaranse datos de calibración de cada unha das articulacións por separado, para calcular unha transformación específica para cada unha delas. Así, obterase un conxunto de puntos diferente por articulación, illando os erros e proporcionando a posibilidade de tratalos por separado.

O resultado desta etapa de recollida de datos de calibración son dous conxuntos de puntos por articulación, un procedente de cada dispositivo, cos que se realizará

o cálculo da transformación necesaria para cada articulación.

6.3.2. Transformación lineal - Transformación afín

Para conseguir que os dous dispositivos traballen no mesmo sistema de coordenadas é necesario aplicarlle unha transformación aos datos capturados. Consideráronse dous tipos de transformación [38]:

Lineal: Na que se teñen en conta a rotación e translación que é necesario aplicarlle ao sistema de coordenadas dun dos dispositivos para que se sitúe exactamente no mesmo lugar e coa mesma orientación que o do outro.

Afín: Na que, ademais de ter en conta a rotación e translación como na transformación *lineal*, tamén se consideran a escala, o estiramento ou *shear* e a reflexión.

Como xa se comentou na sección anterior, existe un erro inherente a cada dispositivo que produce diferencias na posición capturada para un determinado punto con respecto a outros dispositivos. A transformación *lineal* só considera a rotación e translación do conxunto de puntos, pero mantendo intactas as posicións relativas entre os distintos puntos de dito conxunto. Tendo en conta o erro debido a cada dispositivo é moi probable que as posicións relativas entre os puntos nos conxuntos recollidos por dous dispositivos diferentes, non sexa a mesma polo que unha transformación *lineal* non sería adecuada e queda descartada.

A transformación *afín*, pola súa banda, si ten en conta estas variacións nas posicións relativas entre puntos dos dous conxuntos a considerar. Este tipo de transformación inclúe deformacións de escala, estiramento e reflexión sobre o conxunto de puntos, contemplando a discrepancia nas diferencias relativas entre os puntos capturados por dispositivos diferentes e reducindo aínda máis o erro debido ás diferencias na captura. Ademais, é necesario ter en conta que unha transformación *lineal* sempre é *afín*, pero unha transformación *afín* non ten porque ser *lineal*, polo que aplicando a transformación *afín* vai incluida tamén a *lineal*. Por estas razóns, a transformación *afín* é a escollida para calibrar o sistema.

Aínda que existen múltiples implementacións e algoritmos para o cálculo da transformación *afín* partindo de dous conxuntos de puntos correspondentes, a maioría son en dúas dimensións, é dicir, no plano, e non son facilmente extrapolables ao espazo. Despois dun longo proceso de busca atopouse unha implementación do cálculo da transformación *afín* no espazo polo método de mínimos cadrados, codificado

en *Matlab*, e incluido na libraría **iso2mesh**, desenvolvida e publicada en 2008 con licencia *GPL v2* por *Q. Fang* [39]. Este algoritmo do cálculo da transformación *afín* foi adaptado á linguaxe de implementación do subsistema Servidor, *C#*, coa axuda da libraría *Math.Net* [40], para posteriormente incluílo no módulo de calibración.

O resultado do cálculo da transformación *afín* con este algoritmo é, por unha banda, unha matriz de 3x3, onde se inclúen a rotación, escala, estiramento e reflexión, e por outra banda, un vector de translación de lonxitude 3. Estes dous elementos conforman a transformación *afín*.

A transformación calculada aplícase sobre os datos dun dos dispositivos para que traballe no mesmo sistema de coordenadas que o outro. Para aplicar esta transformación realízanse os seguintes cálculos sobre os datos de captura:

$$PuntoTransformado = M * PuntoReal + v$$

M = Matriz 3x3 resultado do cálculo da transformación afín

PuntoReal = Posición dun punto no espazo capturada por un dispositivo. Este punto será tratado como un vector de 3x1 para poder multiplicalo pola matriz de transformación de 3x3.

v = Vector de translación 3x1 resultado do cálculo da transformación afín

PuntoTransformado = Posición do PuntoReal no sistema de coordenadas unha vez se lle aplicou a tranformación lineal

6.3.3. Interpolación

Como a transformación *afín* aplicada aos dispositivos ven cun erro dependente do hardware difícil de eliminar, é necesario interpolar nas transicións entre os datos de captura dun dispositivo e doutro. Desta maneira, para acadar os datos de captura finais no subsistema Servidor é necesario establecer unha xerarquía de dispositivos de captura. A xerarquía de dispositivos resulta útil para evitar ter que realizar cálculos de interpolación entre os datos para o mesmo óso durante toda a captura, realizando estes cálculos só cando un dispositivo deixa de captar algún óso e cando volve a captalo.

Desta maneira traballarase cun *Kinect* principal, que será do que se tomen os datos de captura ata que se produza unha oclusión que non permita ver algún óso.

Nese momento o sistema comproba se ese óso é captado polo segundo *Kinect*, e se é así, realiza unha transición dos datos do *Kinect* principal aos do *Kinect* secundario. En caso de que o óso volva a ser captado polo primeiro *Kinect*, realízase unha transición de novo dos datos do *Kinect* secundario ao principal.

O principal problema destas transicións dáse no caso de ósos que sufran un erro grande ao realizar a transformación *afín*, de forma que entre o valor de posición do *Kinect* principal e o do secundario transformado existe unha distancia apreciable. Se a transición se realiza de maneira directa da como resultado saltos tal e como se mostra na figura 6.4, que se aprecian na animación final.

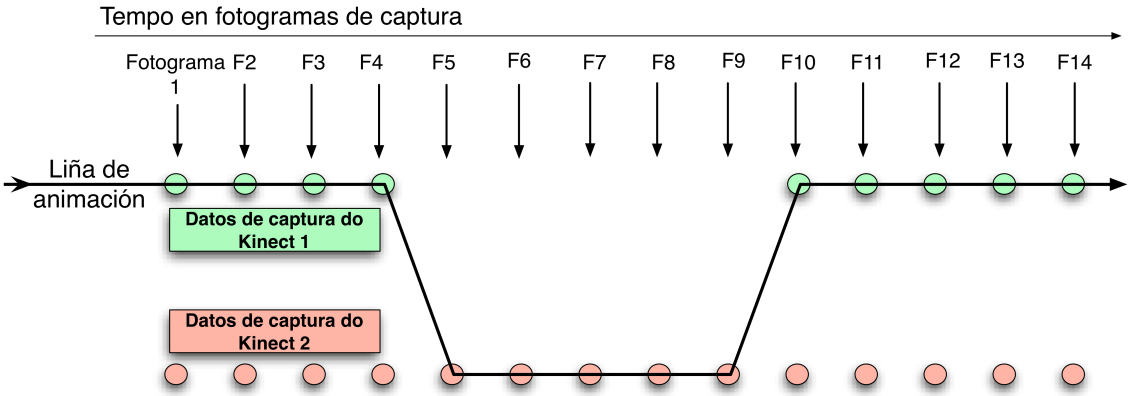

Figura 6.4: Secuencia de fotogramas que mostra o paso dos datos de captura do *Kinect* 1 ao *Kinect* 2 sen interpolación, xerando un salto brusco na animación.

Para evitar isto, elaborouse un sistema de interpolación específico para este caso. Para a construción deste sistema partiuse da función *Lerp* da libraría *XNA* [42]. Esta función recibe tres parámetros, dous puntos no espazo e un valor que denominaremos *ValorIntermedio* e que toma valores entre 0 e 1:

- Se *ValorIntermedio* toma valor 0 devolve o punto 1.
- Se *ValorIntermedio* toma valor 1 devolve o punto 2.
- Os valores entre 0 e 1 de *ValorIntermedio* proporcionan todo o rango de puntos no espazo entre o punto 1 e o 2, sendo 0,5 o punto medio.

A función *Lerp* é o instrumento necesario para realizar a transición no **espazo** entre un punto captado por un *Kinect* e o mesmo punto captado polo outro *Kinect*. Pero é necesario que esta transición se realice tamén ao longo do **tempo** de maneira que o valor de interpolación irá achegándose desde un punto a outro de maneira progresiva. Para isto introduciuse o concepto de **Granularidade**, que indica o número de fotogramas nos que se debe realizar a transición.

Así, coa función *Lerp* e coa *Granularidade* conséguese a transición no espazo e no tempo tal como se mostra na figura 6.5.

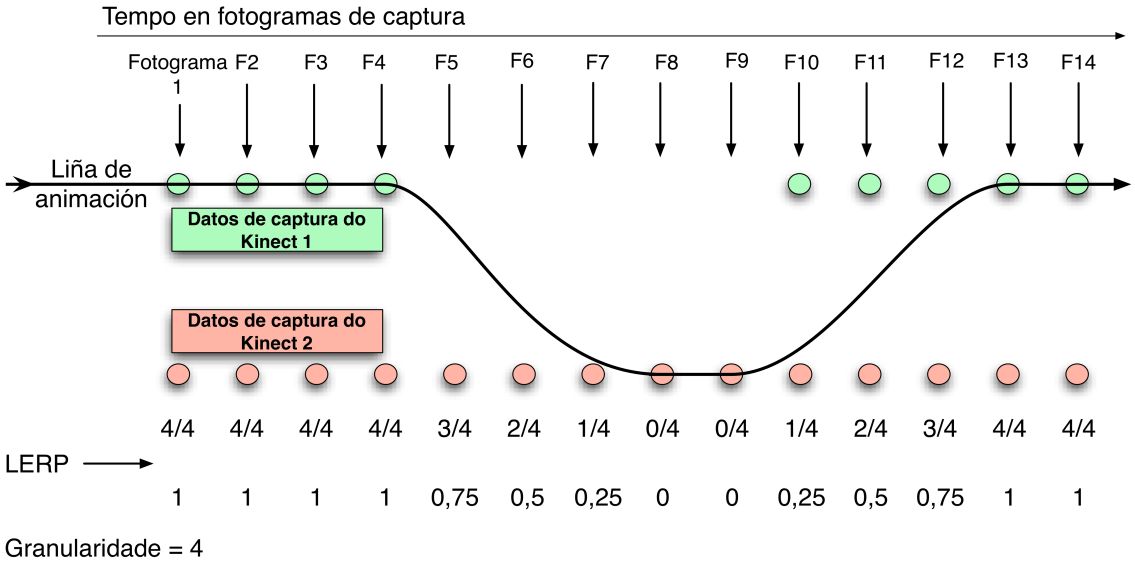

Figura 6.5: Secuencia de fotogramas que mostra o paso dos datos de captura do *Kinect* 1 ao *Kinect* 2 con interpolación, conseguindo unha transición máis suave e polo tanto unha mellor animación.

O valor de *ValorIntermedio* que debe recibir a función *Lerp* calcúlase da seguinte maneira:

$$ValorIntermedio = \frac{ValorTransicion}{Granularidade}$$

ValorTransición comeza co mesmo valor que a *Granularidade*, de maneira que *ValorIntermedio* terá valor 1 e, polo tanto, o sistema traballará cos valores do *Kinect* principal. Tal como se aprecia na figura 6.5, se no *Kinect* principal se produce unha oclusión e deixa de proporcionar datos, *ValorTransicion* vai diminuindo o seu valor en 1 en cada fotograma ata chegar a un valor de 0, de maneira que os puntos proporcionados por *Lerp* se vaian achegando paulatinamente ao valor do *Kinect* secundario.

No caso de que a oclusión cese, *ValorTransición* vai aumentando o seu valor en 1 en cada fotograma ata chegar a un valor igual á *Granularidade*, acadando un achegamento progresivo desde o *Kinect* secundario ao principal.

Por último, se a oclusión cesa nun fotograma sen ter completado a *Granularidade*, ou o que é o mesmo, no medio da progresión entre o dato dun *Kinect* e outro, o *ValorTransición* deixará de diminuír o seu valor para aumentalo progresivamente

ata volver ao dato do *Kinect* principal.

Utilizando este método de interpolación conséguese unha animación final máis natural e progresiva, sen saltos, incluso no caso de que houbese oclusións. Ademais minimízase en grande medida o erro procedente do *hardware* no cálculo da transformación *afín*, xa que aínda que traballemos cunha diferencia duns poucos centímetros entre o valor de posición captado polo *Kinect* principal e o valor transformado captado polo *Kinect* secundario, a transición entre un e outro vai ser, cando menos, máis suave.

Capítulo 7

Probas

Neste capítulo móstranse as probas realizadas ao final de cada *Sprint*. Estas probas foron realizadas en presencia do cliente, e en moitos casos, polo propio cliente, para demostrar que os requisitos especificados para o produto se ían cumprindo durante o desenvolvemento.

7.1. Listado de probas

Proba - Pr.01

Título: Control de conexión dos dispositivos de captura

Requirimentos involucrados: RF.01

Sprint: Sp.05

Descrición:

- O usuario inicia a aplicación servidor con un ou mais dispositivos de captura conectados.

Resultado:

- A aplicación servidor recoñece automaticamente cantos dispositivos hai conectados e indica o número.
- Si o número de dispositivos é alomenos un a aplicación debe permitir realizar unha gravación.
- Si o número de dispositivos é alomenos dous a aplicación debe permitir realizar unha calibración.

Proba - Pr.02

Título: Adquisición e mostra de datos de posición

Requirimentos involucrados: RF.02, RIU.01

Sprint: Sp.06

Descrición:

- O usuario inicia a aplicación servidor cun dispositivo de captura conectado ao equipo.

- O intérprete sitúase diante do dispositivo de captura.

Resultado:

- A aplicación servidor mostra nunha pantalla a imaxe de vídeo recollida polo dispositivo de captura
- A aplicación servidor mostra noutra pantalla a representación das articulacións do actor no espacio.
- Esta representación mostra os movementos do intérprete

Proba - Pr.03

Título: Conexión con *Autodesk Maya*.

Requirimentos involucrados: RF.05, RF.06, RDe.01, RDe.03

Sprint: Sp.07

Descrición:

- O usuario inicia a aplicación servidor cun dispositivo de captura conectado ao equipo
- O usuario inicia *Autodesk Maya*
- O usuario inicia o subsistema Cliente no contorno de *Autodesk Maya*
- O usuario inicia a gravación da sesión de interpretación no subsistema Servidor sen iniciar a captura no subsistema Cliente
- O intérprete sitúase diante do dispositivo de captura e realiza movementos
- Unha vez iniciada a sesión de interpretación o usuario inicia a captura no subsistema Cliente

Resultado:

- O subsistema Cliente no contorno de *Autodesk Maya* mostra os datos de captura que vai recollendo do Servidor en formato texto

Proba - Pr.04

Título: Animación baseada en posicións en *Autodesk Maya*.

Requirimentos involucrados: RF.08, RF.10, RF.12, RF.13, RDe.01, RDe.03

Sprint: Sp.08

Descrición:

- O usuario inicia a aplicación servidor cun dispositivo de captura conectado ao equipo
- O usuario inicia *Autodesk Maya*
- O usuario inicia o subsistema Cliente no contorno de *Autodesk Maya*
- O usuario crea un esqueleto para almacenar unha animación baseada en posicións
- O usuario inicia a gravación da sesión de interpretación no subsistema Servidor sen iniciar a captura no subsistema Cliente
- O intérprete sitúase diante do dispositivo de captura e realiza movementos
- Unha vez iniciada a sesión de interpretación o usuario inicia a captura no subsistema Cliente

Resultado:

- Móstrase a interface de usuario do subsistema Cliente dentro de *Autodesk Maya* coas opcións implementadas ata a data
- O subsistema Cliente recibe os datos desde o principio da captura por parte do Servidor mostrando a existencia da cola de almacenamento intermedio na que se gardan os datos.
- Pódese ver a animación correspondente á sesión de interpretación que se está a realizar en *Autodesk Maya*.
- Ao rematar a sesión de interpretación é posible interactuar coa animación que se acaba de obter en *Autodesk Maya*.

Proba - Pr.05

Título: Animación baseada en rotacións en *Autodesk Maya*.

Requirimentos involucrados: RF.03, RF.09, RF.11, RF.12, RF.14, RDe.01

Sprint: Sp.09

Descrición:

- O usuario inicia a aplicación servidor cun dispositivo de captura conectado ao equipo
- O usuario inicia *Autodesk Maya*
- O usuario inicia o subsistema Cliente no contorno de *Autodesk Maya*
- O usuario crea un esqueleto para almacenar unha animación baseada en rotacións
- O usuario escolle a opción de xerar unha animación baseada en rotacións
- O usuario inicia a gravación da sesión de interpretación no subsistema Servidor sen iniciar a captura no subsistema Cliente
- O intérprete sitúase diante do dispositivo de captura e realiza movementos
- Unha vez iniciada a sesión de interpretación o usuario inicia a captura no subsistema Cliente

Resultado:

- Móstrase a interface de usuario do subsistema Cliente dentro de *Autodesk Maya* coas opcións implementadas ata a data
- O subsistema Cliente recibe os datos desde o principio da captura por parte do Servidor mostrando a existencia da cola de almacenamento intermedio na que se gardan os datos.
- Pódese ver a animación correspondente á sesión de interpretación que se está a realizar en *Autodesk Maya*.
- Ao rematar a sesión de interpretación é posible interactuar coa animación que se acaba de obter en *Autodesk Maya* e pódese comprobar que os ósos do esqueleto involucrado na animación teñen datos de rotación gravados.

Proba - Pr.06

Título: Conexión con *Blender*.

Requirimentos involucrados: RF.05, RF.06, RDe.02

Sprint: Sp.10

Descrición:

- O usuario inicia a aplicación servidor cun dispositivo de captura conectado ao equipo

- O usuario inicia *Blender*
- O usuario inicia o subsistema Cliente no contorno de *Blender*
- O usuario inicia a gravación da sesión de interpretación no subsistema Servidor sen iniciar a captura no subsistema Cliente
- O intérprete sitúase diante do dispositivo de captura e realiza movementos
- Unha vez iniciada a sesión de interpretación o usuario inicia a captura no subsistema Cliente

Resultado:

- O subsistema Cliente no contorno de *Blender* mostra os datos de captura que vai recollendo do Servidor en formato texto

Proba - Pr.07

Título: Animación baseada en rotacións en *Blender*.

Requirimentos involucrados: RF.09, RF.11, RF.12, RF.14, RDe.02

Sprint: Sp.11

Descrición:

- O usuario inicia a aplicación servidor cun dispositivo de captura conectado ao equipo
- O usuario inicia *Blender*
- O usuario inicia o subsistema Cliente no contorno de *Blender*
- O usuario crea un esqueleto para almacenar unha animación baseada en rotacións
- O usuario inicia a gravación da sesión de interpretación no subsistema Servidor sen iniciar a captura no subsistema Cliente
- O intérprete sitúase diante do dispositivo de captura e realiza movementos
- Unha vez iniciada a sesión de interpretación o usuario inicia a captura no subsistema Cliente

Resultado:

- Móstrase a interface de usuario do subsistema Cliente dentro de *Autodesk Maya* coas opcións implementadas ata a data

- O subsistema Cliente recibe os datos desde o principio da captura por parte do Servidor mostrando a existencia da cola de almacenamento intermedio na que se gardan os datos.
- Pódese ver a animación correspondente á sesión de interpretación que se está a realizar en *Blender*.
- Ao rematar a sesión de interpretación é posible interactuar coa animación que se acaba de obter en *Blender* e pódese comprobar que os ósos do esqueleto involucrado na animación teñen datos de rotación gravados.

Proba - Pr.08

Título: Calibración I - Adquisición de datos para calibración

Requirimentos involucrados: RF.15, RF.16, RIU.01

Sprint: Sp.12

Descrición:

- O usuario inicia a aplicación servidor con dous dispositivos de captura conectados ao equipo
- O usuario inicia a gravación da sesión de interpretación no subsistema Servidor
- O intérprete sitúase diante do dispositivo de captura e realiza movementos

Resultado:

- Móstranse dúas ventás, unha asignada para cada dispositivo.
- En cada unha das ventás móstrase unha representación do esqueleto superposto á imaxe do intérprete demostrando que se están a recoller datos dos dous dispositivos á vez en tempo real.

Proba - Pr.09

Título: Calibración II - Cálculo dos parámetros da transformación afín

Requirimentos involucrados: RDe.05

Sprint: Sp.13

Descrición:

- O usuario inicia a aplicación servidor con dous dispositivos de captura conectados ao equipo
- O usuario inicia a gravación da sesión de interpretación no subsistema Servidor
- O intérprete sitúase diante do dispositivo de captura e realiza movementos
- O usuario remata a gravación da sesión de interpretación

Resultado:

- Xenerase un ficheiro no que se mostran os datos de captura recollidos polo *Kinect* principal xunto aos datos de captura obtidos polo Kinect secundario, tanto orixinais como transformados.
- Neste ficheiro pódese ver que, aínda que con certo erro, a aplicación da transformación afín sitúa aos dous *Kinects* no mesmo sistema de coordenadas

Proba - Pr.10

Título: Calibración III - Procesamento dos datos de captura

Requirimentos involucrados: RF.17

Sprint: Sp.14

Descrición:

- O usuario inicia a aplicación servidor con dous dispositivos de captura conectados ao equipo
- O usuario inicia a etapa de calibración no subsistema Servidor
- O intérprete sitúase diante dos dispositivos de captura de maneira que o seu corpo completo é visualizado polos dous a un tempo e realiza a etapa de calibración
- O usuario inicia a gravación da sesión de interpretación no subsistema Servidor
- O intérprete xira o seu corpo ata que un dos seus brazos deixa de ser visualizado polo *Kinect* principal, pero si é visualizado polo *Kinect* secundario.

- Unha vez nesa posición o intérprete move o brazo

Resultado:

- Pódese ver como aínda que o brazo non é visualizado polo *Kinect* principal os movementos son recollidos polo sistema, que fai uso dos datos do *Kinect* secundario transformados, demostrando que a transformación afín funciona.

Proba - Pr.11

Título: Ficheiro de datos de captura

Requirimentos involucrados: RF.07, RF.08, RDe.03, RA.01

Sprint: Sp.15

Descrición:

- O usuario inicia a aplicación servidor con dous dispositivos de captura conectados ao equipo
- O usuario inicia a gravación da sesión de interpretación no subsistema Servidor
- O intérprete sitúase diante do dispositivo de captura e realiza movementos
- O usuario remata a gravación da sesión de interpretación
- O usuario garda a sesión de interpretación nun ficheiro
- O usuario inicia *Autodesk Maya*
- O usuario inicia o subsistema Cliente no contorno de *Autodesk Maya*
- O usuario crea un esqueleto para almacenar unha animación baseada en posicións ou rotacións
- O usuario escolle a opción de xerar unha animación baseada en posicións ou rotacións
- O usuario abre o ficheiro que contén a sesión de interpretación desde o subsistema Cliente

Resultado:

- Unha animación é xerada a partires dos datos recollidos do ficheiro

Proba - Pr.12

Título: Remate do sistema

Requirimentos involucrados: RF.19, RF.20, RF.21, RF.22

Sprint: Sp.16

Proba - Pr.12.1

Descrición:

- O usuario inicia a aplicación servidor con un dispositivo de captura conectado ao equipo
- O usuario di un comando dos predeterminados en voz alta

Resultado:

- O sistema executa este comando

Proba - Pr.12.2

Descrición:

- O usuario inicia a aplicación servidor con un dispositivo de captura conectado ao equipo
- O usuario modifica a inclinación do dispositivo desde o subsistema Servidor

Resultado:

- A inclinación do dispositivo varía en consecuencia.

Proba - Pr.12.3

Descrición:

- O usuario inicia a aplicación servidor cun dispositivo de captura conectado ao equipo
- O usuario inicia *Autodesk Maya*

- O usuario inicia o subsistema Cliente no contorno de *Autodesk Maya*
- O usuario crea un esqueleto para almacenar unha animación baseada en rotacións
- O usuario escolle a opción de xerar unha animación baseada en rotacións
- O usuario escolle a opción de xerar unha animación a partires do fotograma 100
- O usuario escolle a opción de que a personaxe non se desplace no espazo durante a animación
- O usuario inicia a gravación da sesión de interpretación no subsistema Servidor sen iniciar a captura no subsistema Cliente
- O intérprete sitúase diante do dispositivo de captura e realiza movementos
- Unha vez iniciada a sesión de interpretación o usuario inicia a captura no subsistema Cliente

Resultado:

- A animación resultante comeza no fotograma 100, tal como se indicou.
- A personaxe non se traslada no espazo, so mostra unha animación das súas diferentes partes pero sen moverse do sitio.

Capítulo 8

Conclusións e traballo futuro

8.1. Conclusións

O obxectivo deste traballo fin de grao foi o desenvolvemento dun sistema de **baixo custo** para a xeración de animacións a partir da captura de movemento sen marcas dunha sesión de interpretación realizada por un actor humano. O traballo xustifícase no interese de proporcionar, a pequenas empresas e traballadores por conta propia do mundo da animación por ordenador, un entorno de traballo que lles permita o desenvolvemento de animacións sen ter que desembolsar as cantidades de diñeiro que as grandes empresas solicitan polos seus produtos.

O resultado deste proxecto é un sistema que cumpre cos obxectivos propostos e satisface todos os requisitos designados polo cliente:

- O sistema permite a xeración de **animacións en tempo real**, mediante as aplicacións *Autodesk Maya* e *Blender*, partindo dos datos de captura de movemento procedentes dun dispositivo de baixo custo, concretamente o *Kinect* de Microsoft.
- O sistema permite a captura simultánea de datos desde **dous dispositivos**, aplicándolles a necesaria transformación para que a representación se realice no mesmo sistema de coordenadas, conseguindo minimizar deste xeito as oclusións que se producen na utilización dun solo sistema de captura.
- O sistema permite o **almacenamento** en ficheiro dos datos de captura procedentes da gravación dunha sesión de interpretación, e o posterior uso deste para xerar unha animación.

Ademais destas características básicas que aseguran o cumprimento dos obxectivos propostos, debemos poñer tamén de manifesto un conxunto de solucións destinadas a mellorar a calidade global da solución proporcionada. Neste sentido:

- Cabe salientar que o feito de que para a **calibración** do sistema non sexa necesario ningún elemento externo, e se poida facer co propio corpo do intérprete, proporciona unha maior facilidade de uso do sistema.
- Este proxecto proporciona unha nova perspectiva ao uso do *Kinect*, un dispositivo inicialmente concibido para o mundo dos videoxogos. O feito de desenvolver un sistema que consegue coordinar **dous *Kinects*** para obter unha animación en **tempo real**, nun software de xeración de gráficos por ordenador, supuxo un reto tecnolóxico que, unha vez superado, permite ampliar as

funcionalidades do dispositivo e, xa que logo, unha adopción máis ambiciosa desde o entorno profesional.

- A metodoloxía escollida, *Scrum*, resultou completamente acertada para o desenvolvemento deste proxecto: permitiu unha interacción áxil co cliente, que estivo involucrado durante todo o proxecto, facilitando un proceso de validación continua. Este feito tivo como resultado unha evolución nos requisitos, que se foron adaptando, iteración a iteración, ao que o cliente necesitaba, obtendo un produto final moito máis acorde co que el esperaba. Para o equipo de desenvolvemento tamén foi moi beneficioso, xa que ter ese retorno continuado do cliente permite xestionar dunha maneira máis eficaz a resolución de problemas e a xestión de continxencias.

- En canto ao deseño das diferentes partes do sistema, as solucións escollidas reflicten un baixo acoplamento entre as distintas responsabilidades, permitindo a súa reutilización por separado e proporcionando unha base sólida para o traballo futuro.

8.2. Traballo futuro

En xeral, a fase de captura de requisitos permite formalizar a evolución dun proxecto máis aló do prazo de que se dispón para a realización dunha primeira versión. Resulta convinte neste sentido diferenciar claramente as prioridades dos requisitos que forman parte da análise, de xeito que os requisitos opcionais permitan configurar o previsible futuro do proxecto. A continuación facemos un resumo daqueles requisitos que forman os seguintes pasos a realizar, unha vez finalizado o presente proxecto:

- A internacionalización da aplicación, que se realizou en lingua inglesa, e á que o cliente lle gustaría permitir estender a outros idiomas.

- A liberación do código con licenza de software libre. Para iso será necesario o estudo das diferentes licenzas e a avaliación da viabilidade polo uso de certas librarías que non son de software libre.

Por outra banda, a *SDK* que *Microsoft* ten dispoñible para a interacción con *Kinect* proporciona unha serie de funcionalidades que, non sendo necesarias neste proxecto, sería interesante consideralas para futuras versións, por exemplo:

- A posibilidade de conectar ata catro dispositivos a un tempo.
- A posibilidade de identificar ata catro esqueletos ao mesmo tempo nunha escena.

Outra ampliación interesante sería adaptar a parte cliente para outras aplicacións software de xeración de gráficos por ordenador, como son *3DStudio Max*, *Softimage* ou *Cinema4D*.

Anexo A: Manual de usuario

O primeiro paso para poder utilizar o sistema é conectar ao equipo un ou dous dispositivos de captura *Kinect*.

Gravación dunha sesión de interpretación cun só dispositivo

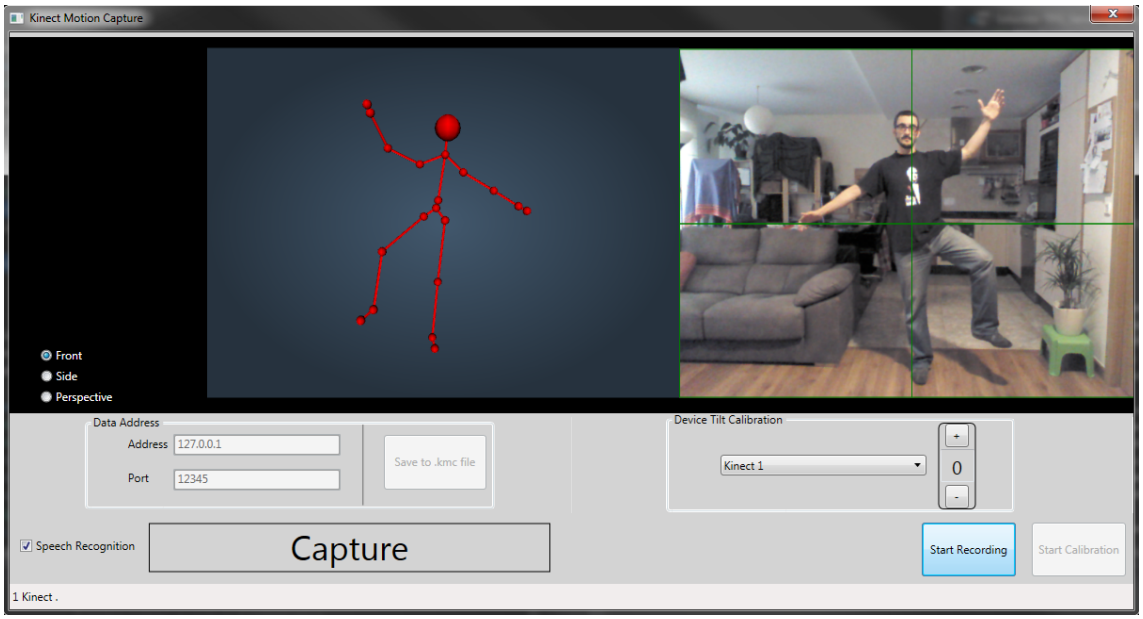

Figura 8.1: Ventá da aplicación durante unha sesión de interpretación

1. Iniciar a aplicación. Debería mostrarse unha ventá como a da figura 8.1.
2. Situarse diante do dispositivo de captura.
3. Pulsar o botón *Start recording*, ou dicir en voz alta a palabra *"Capturar"* e con iso comezará a gravación da sesión de interpretación.

4. Realizar a sesión de interpretación tentando non saír da zona de captura.

5. Para rematar a gravación volver a pulsar o botón *Start recording*, ou dicir en voz alta a palabra textit“Acabar“.

Gravación dunha sesión de interpretación cun dous dispositivos

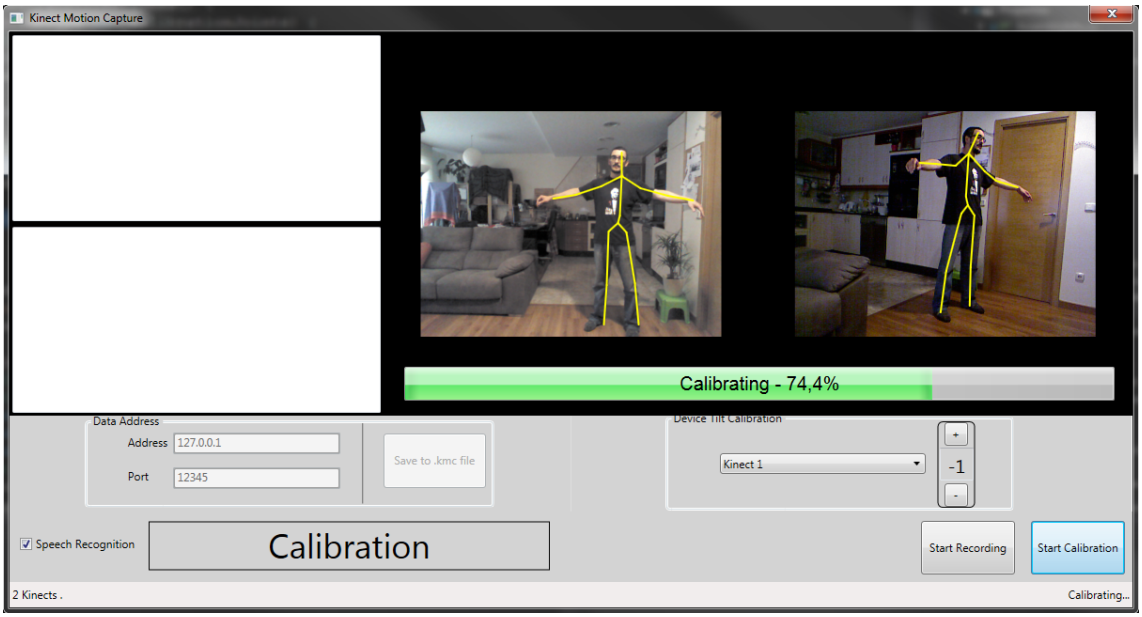

Figura 8.2: Ventá da aplicación durante a etapa de calibración

1. Iniciar a aplicación. Debería mostrarse unha ventá como a da figura 8.2.

2. Situarse diante dos dispositivos de captura de maneira que o corpo completo do suxeito sexa visualizado por ámbolos dous.

3. Pulsar o botón *Start calibration*, ou dicir en voz alta a palabra *“Calibrar“* e con iso comezará a etapa de calibración do sistema.

4. O suxeito debe moverse por toda a zona de captura tentando que o seu corpo completo sexa capturado por ambos dispositivos. Cando algunha parte do suxeito non é capturada polo sistema o esqueleto superposto á imaxe queda quieto; unha vez se recupera a captura de todo o corpo o esqueleto volve a poñerse en movemento acorde co corpo do suxeito. So se recollerán datos de

calibración cando o corpo do suxeito sexa visualizado por ambos dispositivos. O progreso de recollida destes pode observarse na barra debaixo da visualización de vídeo.

5. Cando a barra de progreso chega ao 100 % significa que a etapa de calibración rematou.

6. O sistema carga o módulo que permite realizar a sesión de interpretación.Debería mostrarse unha ventá como a da figura 8.1.

7. Situarse diante do dispositivo de captura.

8. Pulsar o botón *Start recording*, ou dicir en voz alta a palabra *"Capturar"* e con iso comezará a gravación da sesión de interpretación.

9. Realizar a sesión de interpretación tentando non saír da zona de captura.

10. Para rematar a gravación volver a pulsar o botón *Start recording*, ou dicir en voz alta a palabra textit"Acabar".

Xeración de animación en Autodesk Maya

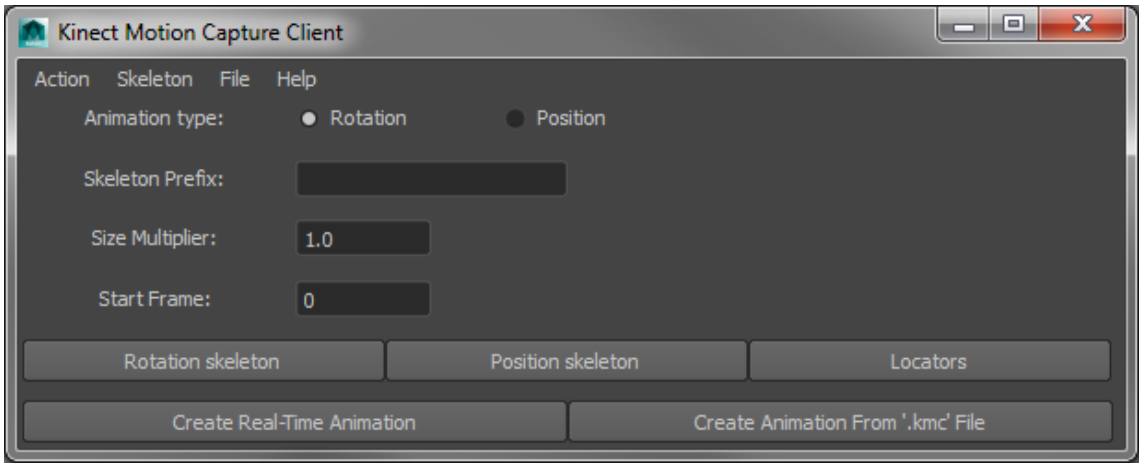

Figura 8.3: Ventá do módulo de captura de movemento con *Kinect* no contorno de *Autodesk Maya*

1. Iniciar a aplicación servidor. Debería mostrarse unha ventá como a da figura 8.1.

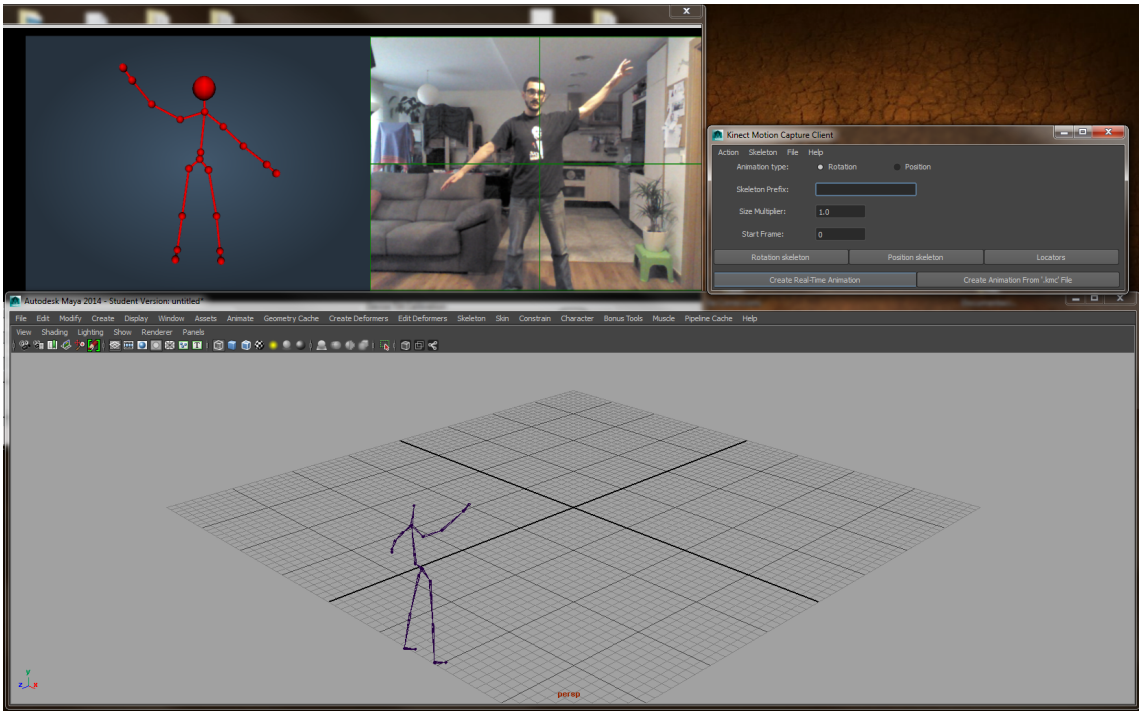

Figura 8.4: Ventá do módulo de captura de movemento con *Kinect* no contorno de *Autodesk Maya*

2. Iniciar *Autodesk Maya*.

3. Iniciar o módulo de captura de movemento con *Kinect*. Debería mostrarse unha ventá como a da figura 8.3.

4. Escoller animación baseada en rotacións ou en posicións.

5. Crear o *Skeleton* segundo a elección anterior no botón *Rotation skeleton* ou *Position skeleton*.

6. Pulsar o botón *Create Real-Time Animation*

7. Situarse diante do dispositivo de captura.

8. No servidor pulsar o botón *Start recording*, ou dicir en voz alta a palabra *"Capturar"* e con iso comezará a gravación da sesión de interpretación.

9. Realizar a sesión de interpretación tentando non saír da zona de captura. Os movementos do suxeito deben verse reflexados no *Skeleton* creado en *Maya* tal como se mostra na figura 8.4.

10. Para rematar a gravación volver a pulsar o botón *Start recording*, ou dicir en voz alta a palabra textit“Acabar“.

Xeración de animación en Blender

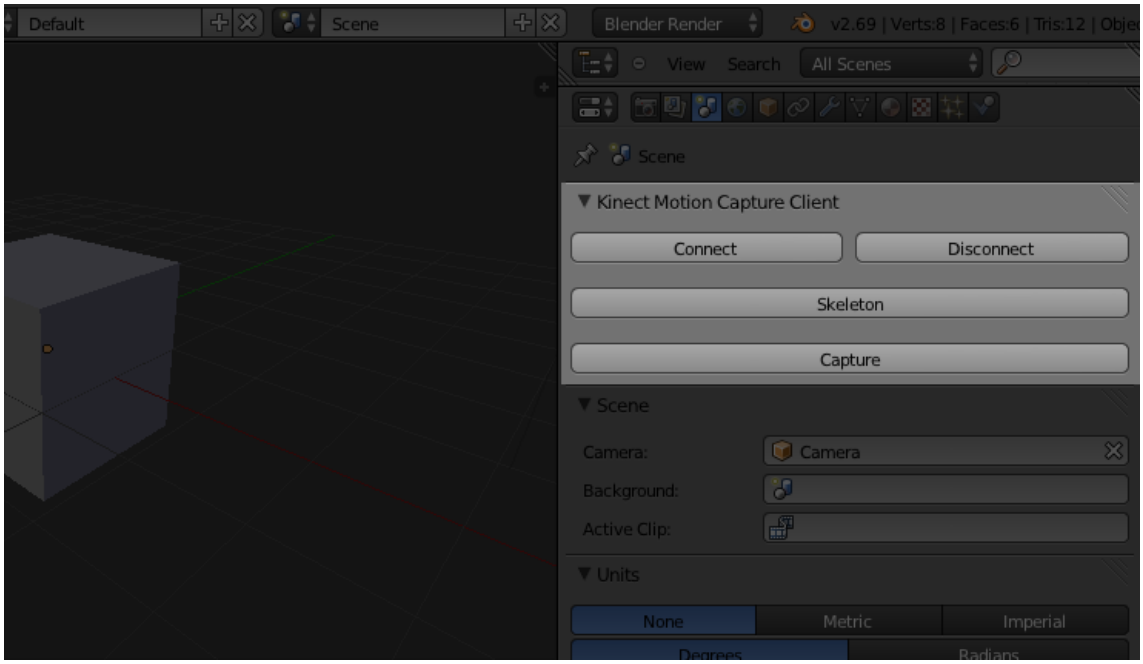

Figura 8.5: Ventá do módulo de captura de movemento con *Kinect* no contorno de *Blender*

1. Iniciar a aplicación servidor. Debería mostrarse unha ventá como a da figura 8.1.
2. Iniciar *Blender*.
3. Iniciar o módulo de captura de movemento con *Kinect*. Debería mostrarse unha ventá como a da figura 8.5.
4. Pulsar o botón *Connect*.
5. Crear o *Skeleton* pulsando o botón *Skeleton*.
6. Pulsar o botón *Capture*
7. Situarse diante do dispositivo de captura.

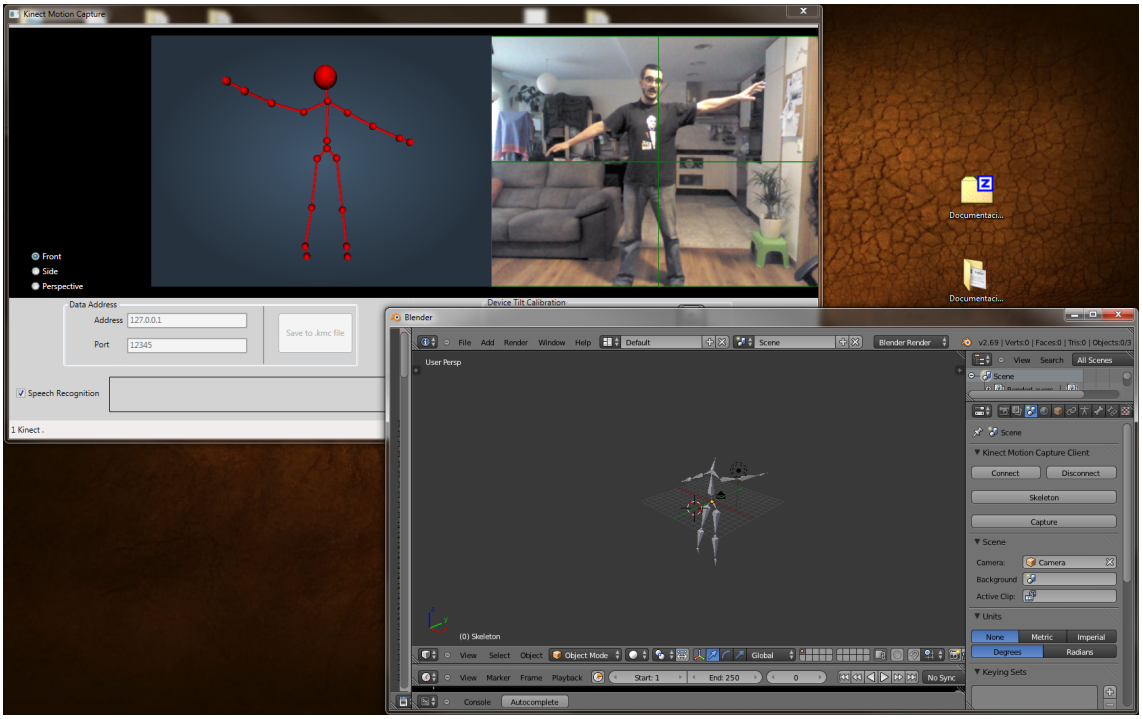

Figura 8.6: Ventá do módulo de captura de movemento con *Kinect* no contorno de *Blender*

8. No servidor pulsar o botón *Start recording*, ou dicir en voz alta a palabra *"Capturar"* e con iso comezará a gravación da sesión de interpretación.

9. Realizar a sesión de interpretación tentando non saír da zona de captura. Os movementos do suxeito deben verse reflexados no *Skeleton* creado en *Blender* tal como se mostra na figura 8.6.

10. Para rematar a gravación volver a pulsar o botón *Start recording*, ou dicir en voz alta a palabra textit"Acabar".

Gravación dunha sesión de interpretación a un ficheiro

1. Iniciar a aplicación. Debería mostrarse unha ventá como a da figura 8.1.

2. Situarse diante do dispositivo de captura.

3. Pulsar o botón *Start recording*, ou dicir en voz alta a palabra *"Capturar"* e con iso comezará a gravación da sesión de interpretación.

4. Realizar a sesión de interpretación tentando non saír da zona de captura.
5. Para rematar a gravación volver a pulsar o botón *Start recording*, ou dicir en voz alta a palabra textit“Acabar“.
6. Pulsar o botón *Save to .kmc file*
7. Escoller o nome do ficheiro e salvalo

Creación de animación desde ficheiro

1. Iniciar *Autodesk Maya*.
2. Iniciar o módulo de captura de movemento con *Kinect*. Debería mostrarse unha ventá como a da figura 8.3.
3. Escoller animación baseada en rotacións ou en posicións.
4. Crear o *Skeleton* segundo a elección anterior no botón *Rotation skeleton* ou *Position skeleton*.
5. Pulsar o botón *Create Animation From '.kmc' File*
6. Escoller un ficheiro que conteña unha sesión de interpretación gravada.
7. Unha vez cargado o ficheiro debe observarse como a animación está sendo transferida ao *Skeleton*.

Anexo B: Glosario de termos

Sesión de interpretación: Unha sesión de interpretación consiste nun intérprete que realiza unha serie de movementos diante dun dispositivo ou dispositivos de captura, coa finalidade de trasladar ditos movementos a un software de xeración de gráficos por ordenador, no que se traballará con eles para dotar de animación a unha personaxe.

Animación: Movemento xerado de maneira artificial que realiza un personaxe dentro dun software de creación de gráficos por ordenador.

Captura de movemento sen marcas: A captura de movemento consiste na recollida de datos de posición ou rotación nuns determinados puntos dun corpo en movemento, e se ademais é sen marcas, a recollida destes datos non require as citadas marcas nos puntos de control da animación, xa que isto se fai de forma automática.

Kinect: Dispositivo de captura de movemento sen marcas da casa *Microsoft*. Conta cunha cámara de imaxe real, e unha cámara e un emisor de infravermellos, cos que é capaz de recoñecer e seguir o movemento de varios esqueletos de forma humana adulta e proporcionar os datos de animación.

Autodesk Maya: Paquete software propietario, da casa *Autodesk*, adicado ao desenvolvemento de gráficos tridimensionais por ordenador.

Blender: Paquete software de código aberto adicado ao desenvolvemento de gráficos tridimensionais por ordenador.

Esqueleto: Entidade, formada por ósos e articulacións, nos software de xeración de gráficos por ordenador onde se almacenarán os datos de posición ou rotación recollidos polo dispositivo de captura. Unha vez no esqueleto estes datos conforman unha animación. O esqueleto está organizado de maneira xe-

rárquica, de forma que o movemento daqueles osos que se atopan mais arriba na xerarquía transmítese aos que están mais abaixo.

Óso: Partes que conforman o esqueleto.

Articulación: Parte do óso onde este se une ao que está por encima na xerarquía.

Prototipo vertical Prototipo que se constrúe para probar unha funcionalidade complexa que o equipo de desenvolvemento non entende de maneira clara, e que necesita ser explorada de maneira pragmática.

Algoritmo: Conxunto ordenado e finito de operacións que permite atopar a solución dun problema.

Animación baseada en posicións: Animación que traballa con valores absolutos da posición no espazo de cada óso.

Animación baseada en rotacións: Animación que traballa con valores absolutos da posición no espazo de cada óso.

Animación en tempo real: Tipo de animación que se crea no software de xeración de gráficos por ordenador ao mesmo tempo que está tendo lugar a gravación da sesión de interpretación.

Estudio de Animación: Parte do sistema no que ten lugar a xeración da animación, que transcorre no ámbito dun software de xeración de gráficos por ordenador.

Estudio de Gravación: Parte do sistema na que ten lugar a gravación da sesión de interpretación. Fotograma: cada unha das imaxes individuais que se suceden nunha animación ou nunha película cinematográfica.

Marcas: Pequenos obxectos construídos cun material determinado que os dispositivos de captura son capaces de distinguir do resto da escena. Obtendo as súas posicións no espazo ao longo do tempo permiten construír unha animación.

Modelo poligonal: Modelo que consiste nunha malla de polígonos que se pode deformar e que se utiliza de base para realizar animacións por ordenador.

Oclusión: Perda de datos de captura que ten lugar no dispositivo de captura de movemento cando o intérprete, ao realizar determinados movementos, oculta ao dispositivo certas partes do seu corpo.

Bibliografía

[1] **R. Williams**. *The animator's survival kit*. Ed. Faber and Faber Limited. 2001.

[2] http://thewaltdisneycompany.com/ , *Data de consulta:* Febreiro 2014

[3] http://www.filmsite.org/visualeffects.html , *Data de consulta:* Febreiro 2014

[4] http://trolljack.blogspot.com.es/2011/05/vivi-3d-model-new.html, *Data de consulta:* Febreiro 2014

[5] Ilustración de Heather Jones para TIME. http://entertainment.time.com/2012/12/05/gollums-getup-how-the-hobbits-groundbreaking-technology-works/, *Data de consulta:* Febreiro 2014

[6] http://www.xbox.com/es-ES/xbox-360/why-xbox-360, *Data de consulta:* Febreiro 2014

[7] https://en.wikipedia.org/wiki/Motion_capture, *Data de consulta:* Febreiro 2014

[8] http://www.vicon.com/, *Data de consulta:* Febreiro de 2014

[9] http://www.qualisys.com/applications/media-and-entertainment/ , *Data de consulta:* Febreiro 2014

[10] https://www.naturalpoint.com/optitrack/ , *Data de consulta:* Febreiro 2014

[11] http://www.xsens.com/, *Data de consulta:* Febreiro 2014

[12] http://www.synertial.com/, *Data de consulta:* Febreiro 2014

[13] http://www.xcitex.com/procapture-motion-capture-systems.php, *Data de consulta:* Febreiro 2014

[14] http://www.organicmotion.com/mocap-for-animation/, *Data de consulta:* Maio 2014

[15] http://www.warnerbros.com/, *Data de consulta:* Febreiro 2014

[16] http://www.ubi.com/ES/, *Data de consulta:* Febreiro 2014

[17] http://mocap.cguse.com/en/products.html, *Data de consulta:* Maio 2014

[18] http://ipisoft.com/, *Data de consulta:* Febreiro 2014

[19] http://www.ni-mate.com/, *Data de consulta:* Maio 2014

[20] http://publications.lib.chalmers.se/records/fulltext/156763.pdf, *Data de consulta:* Febreiro 2014

[21] http://www.autodesk.es/products/autodesk-maya/, *Data de consulta:* Febreiro 2014

[22] http://www.blender.org/, *Data de consulta:* Febreiro 2014

[23] http://www.autodesk.com/products/motionbuilder/, *Data de consulta:* Febreiro 2014

[24] **Project Management Institute**. *Guía de los Fundamentos de la Dirección de Proyectos (PMBOK)*

[25] **R. Noya, P. Félix, M.J. Carreira**. *Titulo*. Documento interno. Novembro 2013.

[26] http://plandecarrera.infojobs.net/puesto-de-trabajo/analista%20programador, *Data de consulta:* Xuño 2014

[27] http://www.seg-social.es/Internet_1/Trabajadores/CotizacionRecaudaci10777/Basesytiposdecotiza36537/index.htm, *Data de consulta:* Xuño 2014

[28] http://www.actibva.com/magazine/mercado-laboral/economia-para-todos-el-coste-de-contratacion-de-un-trabajador-para-una-empresa, *Data de consulta:* Xuño 2014

[29] http://www.primesense.com/, *Data de consulta:* Febreiro 2014

[30] https://en.wikipedia.org/wiki/Wii_Remote, *Data de consulta:* Febreiro 2014

[31] https://en.wikipedia.org/wiki/PlayStation_Eye, *Data de consulta:* Febreiro 2014

[32] http://www.asus.com/es/Multimedia/Xtion_PRO_LIVE/, *Data de consulta:* Febreiro 2014

[33] ~~http://www.openni.org~~, *Data de consulta:* Febreiro 2014;(**Enlace roto**) **Actualización**: https://en.wikipedia.org/wiki/OpenNI, *Data de consulta:* Xuño 2014

[34] http://projects.ict.usc.edu/mxr/faast/, *Data de consulta:* Febreiro 2014

[35] **R.C. Martin**. *Agile Principles, Patterns and Practices in C#*. Editorial Prentice Hall. 2007.

[36] **A. Jana**. *Kinect for Windows SDK Programming Guide*. Editorial Packt. 2012.

[37] http://www.usabilityfirst.com/glossary/horizontal-and-vertical-prototypes/, *Data de consulta:* Marzo 2014

[38] **B. Cyganek, J.P. Siebert**. *An Introduction to 3D Computer Vision Techniques and Algorithms*. Editorial Wiley. 2009.

[39] http://sourceforge.net/projects/iso2mesh/files/metch/0.5.0/, *Data de consulta:* Maio 2014

[40] http://www.mathdotnet.com/, *Data de consulta:* Maio 2014

[41] http://www.amo.net/NT/02-21-01FPS.html, *Data de consulta:* Marzo 2014

[42] http://msdn.microsoft.com/en-us/library/microsoft.xna.framework.mathhelper.lerp.aspx, *Data de consulta:* Maio 2014