**UNIVERSIDAD TECONOLÓGICA NACIONAL FACULTAD REGIONAL CÓRDOBA**

**Proyecto Final Optical Marketing**

**Documentación de Desarrollo de Proyecto**

**Grupo 4 Carlos Kapica 51482 Rodrigo Liberal 51658 Julián Peker 51395**

**Fernández David 53063**



# 

Contenido

[Introducción 3](#_Toc371178519)

[Diagrama de Paquetes 4](#_Toc371178520)

[Diagrama de Clase Del Paquete ColorImageStream 6](#_Toc371178521)

[Diagrama de Clases del paquete Capture 7](#_Toc371178522)

[Diagrama de clases del paquete FileSystem 9](#_Toc371178523)

[Diagrama de Clase del paquete Connection 10](#_Toc371178524)

[Diagrama de Clase del paquete HTTPConnection 11](#_Toc371178525)

[Diagrama de clases del paquete Interaction 12](#_Toc371178526)

[Diagrama de Clases del paquete Preprocessing 14](#_Toc371178527)

[Diagrama de Clases del paquete Segmentation 16](#_Toc371178528)

[Diagrama de Clase del paquete Skeleton 17](#_Toc371178529)

[Diagrama de clases del paquete Web 18](#_Toc371178530)

[Diagrama de clases del paquete Monitoring 20](#_Toc371178531)

[Diagrama de clases del paquete Catalog 21](#_Toc371178532)

[Diagrama de clases del paquete GameMemory 22](#_Toc371178533)

[Diagrama de estado de KinectStateMachine 24](#_Toc371178534)

[Diagrama de Estado de AdvertStateMachine 26](#_Toc371178535)

[Diagrama de Estado de Skeleton 28](#_Toc371178536)

[Diagrama de Estado de Catálogo 29](#_Toc371178537)

[Diagrama de Estado de Juego Game Memory 31](#_Toc371178538)

[Diagrama de Despliegue de OpticalMarketing 33](#_Toc371178539)

[Diagrama Entidad Relación de bases de datos relacional 35](#_Toc371178540)

# Introducción

A continuación se presentarán los principales módulos y avances del modelado del proyecto Optical Marketing.

Se incluyen, diagramas entidad relación, diagramas de clases, diagramas de paquetes, diagramas de estado y diagrama de despliegue. Con dichos diagramas se pretende dar una visión global de la arquitectura del proyecto y el funcionamiento del mismo.

# Diagrama de Paquetes

Los paquetes Principales del OpticalMarketing :

* InteractionPackage: paquete dedicado a gestionar las interacciones del Kinect con usuarios que interactúan.
* CapturePackage: paquete dedicado a realizar la captura de imágenes empleando el Kinect.
* CatalogPackage: paquete que contiene la funcionalidad de aplicación de catálogo.
* MonitoringPackage: paquete que contiene la funcionalidad del servicio de monitoreo.
* GameMemoryPackage: paquete que contiene la funcionalidad de aplicación de juego.
* PreprocessingPackage: paquete para realizar las etapas de preprosesamiento de imágenes (aplicación de filtros y conversión de imágenes a matrices).
* SkeletonPackage: paquete empleado para representar el esqueleto humano y sus articulaciones para poder realizar interacciones con el Kinect.
* WebPackage: paquete empleado para el acceso web, muestra de estadistícas y carga de catálogos.

(Si bien existen más paquetes, los mencionados conforman el principal conjunto de elementos necesarios para la captura, análisis y proceso de imágenes y datos del sistema).

Dichos paquetes se relacionan de la siguiente forma: el paquete de InteractionPackage define el comportamiento de cómo se va realizar la interacción entre las personas y la publicidad, tiene acceso sobre el CapturePackage, el cual posee el comportamiento de capturar imágenes, y detección movimientos, generando un cambio de estado, para que el SkeletonPackage pueda detectar personas, gestos, posturas y movimientos.

Para poder realizar la interacción efectivamente se emplearán los Streams (Flujos de Datos) definidos en Skeleton. Se define el CatalogPackage para implementar la interacción con el catálogo de productos. La interacción con el juego se efectúa con el paquete GameMemory.

El Monitoreo del flujo de persona está definido con el paquete MonitoringPackage , este paquete procesa los datos que le proveen los paquetes CapturePackage y SkeletonPackage, para poder realizar el conteo de persona que pasan a una determinada hora en un puesto.

Por otra parte para realizar el filtrado de imágenes capturadas para extraer los elementos y partes significativas, se empleará el PreprocessingPackage. El PreprocessingPackage de imágenes que filtra las imágenes controla al SegmentationPackage como se realizara este comportamiento. La Segmentación separa la imagen en distinta áreas en función de las características que tenga esa parte de la imagen, para ser tratado de la forma correcta.

Para almacenar y visualizar los resultados de captura y procesamiento se emplea al WebPackage.

La comunicación entre el WebPackage y el InteractionPackage se realiza mediante Paquete de conexión que se encarga de realizar el nexo entre ambos paquetes, manipulando el Sistema de Archivos (FileSystem) y la Base de Datos, este paquete es denomidado FileSystemPackage. En el siguiente Diagrama figuran los paquetes principales y de soporte. Luego en los siguientes apartados de este documento se encuentran explicados y descriptos cada uno de los paquetes.



# Diagrama de Clase Del Paquete ColorImageStream

Para ejecutar la captura de una imagen con el sensor Kinect, se definen dos clases: ColorImageStream y ColorImageFrame. La ColorImageStream, se comporta a bajo nivel con el sensor Kinect definiendo cada cuánto tiempo va tomar frames(cuadros de imágenes), resolución de captura, con que foco, el nivel de luz en función de estos parámetros, realizando esta captura con la cámara RGB(Cámara que posee el sensor Kinect), y los encapsula dentro de un Stream (flujo de datos) para luego ser Procesado y definido por la clase ColorImageFrame, esta clase transforma ese flujo de datos en una imagen, definiendo el formato : la cantidad de pixeles, el tamaño de la imagen(alto,ancho).



# Diagrama de Clases del paquete Capture

En este diagrama se define el comportamiento que necesita para Iniciar la captura de datos (imagen o video).Esta basado con el sensor Kinect.

Destacamos en este diagrama, tres clases importantes. Primero la interface Device, con esta interface, se describe que propiedades y que estado posee los dispositivos que se utilicen para la captura, se necesita conocer si está conectado el dispositivo, que acciones tomar cuando se conecta el dispositivo y que acciones tomar cuando se desconecta el dispositivo.

Luego se encuentra clase Kinect, que hereda de Device, además de lo mencionado anteriormente, es importante identificar la Kinect por el hecho que en una aplicación puede haber más de un sensor Kinect conectado, por eso posee un id, también de cada uno de ellos, conocer su estado, en caso de algún error de poder informarlo para que pueda ser tratado, ya que el dispositivo Kinect opera como una máquina de estado, además se identifica cada uno de los controladores(RGBCamera, Microphone,IRProyector, IRCamera, Motor, Accelerometer).

Por último tenemos Capture, que va describir que comportamiento se va lograr cuando se inicie la captura y qué hacer con ese flujo de datos (Stream): definir el formato, en función de estos se definieron dos tipos de captura video e imagen.



# Diagrama de clases del paquete FileSystem

Este Diagrama muestra el comportamiento del acceso al File System (Sistema de archivos) de la terminal donde se corre la aplicación.

Para ellos definimos un controlador (FileSystemManager) que posee una colección de todos los archivos a los cual va acceder o guardar del File System. Cada archivo (FileSystemElement) para ser accedido necesita un atributo elementUrl de FileSystemElement que es la dirección del directorio donde se encuentra el archivo o va ser guardado el archivo. Principalmente el paquete FileSystem es usado para guardar imágenes, pero también piensa destinado a futuro para guardar archivos XML.



# Diagrama de Clase del paquete Connection

En este diagrama se detalla el comportamiento de la interfaz de conexión genérica puede existir entre un AdvertHost y el Servidor Web. Se define una interface de conexión (ConnectionPackage:IConnection), con esto se logra conocer si se efectua la conexión ,como se comporta y los estados de dicha conexión.



# Diagrama de Clase del paquete HTTPConnection

En este diagrama se detalla el comportamiento de conexión entre un AdvertHost y el Servidor Web. Se emplea la interface de conexión (ConnectionPackage:IConnection), con esto se logra conocer si se efectua la conexión ,como se comporta y los estados de dicha conexión.

La HttpConnection es una clase que se utiliza para establecer un dialogo (envío de mensajes http) para conectarse, mandar una solicitud o recibir una repuesta en función de un determinado recurso. Para ello se define un controlador ManagerConnection que va controlar la conexión entre el servidor y el puesto, para resolver las solicitudes que se reciben o envian del Servidor Web.

La funcionalidad principal del paquete HTTPConnection es solicitarle al servidor las imágenes (cargadas por el usuario en una campaña publicitaria) que luego son utilizadas para cualquier advert o propósito que necesite el advert host. En la actualidad se emplea HTTPConnection para descargar imágenes del paquete catalogo.



# Diagrama de clases del paquete Interaction

Este diagrama describe el comportamiento de la interacción de las publicidades.

Se define momentáneamente dos tipos de publicidades: Catalog y InteractiveGame (juego interactivo con fin publicitario). La Clase Advert es la clase fundamental del paquete que tiene definido un tipo de publicidad y tiene asociado un estado. Los distintos estados de las publicidades que se han definido son:

-Closing.

-Disclaiming.

-Handling.

-Monitoring.

-Preparing.

-Running.

-Saving.

Estos estados pueden verse en el apartado del Diagrama de Estado del AdvertStateMachine.

Para la gestión de los estados de los Adverts se utiliza el patrón State que define la interface AdvertState. Se define un controlador AdvertManager, que describe el comportamiento de todas las publicidades en el puesto y determina que publicidad se va ejecutar. En el caso del catalogo cargar las imágenes que muestra por pantalla, ordenar las publicidades, y otras funcionalidades.

Por último para cargar imágenes al catalogo definimos una clase ConnectionAdvertList que tiene

una lista de los distintos elementos, la cual tiene definida la funcionalidad de conectarse con el paquete HTTPConnection para la obtención de recursos del servidor web.



# Diagrama de Clases del paquete Preprocessing

En este diagrama se detalla el preprocesamiento de una imagen aplicando diferentes algoritmos.

Se Definen tres clases importantes: Matrix, Filter y Preprocessor.

La interfaz Filter se utiliza para aplicar distintos tipos de filtro aplicar a la imagen: filtros de bordes, filtro de paso bajo y filtro de paso alto. En el filtro de detección bordes tenemos distintos algoritmos de inteligencia artificial: Sobel, Sharr, Umbral y Canny. En el filtro de paso alto tenemos el algoritmo Laplaciano. Por último en el filtro de paso bajo tenemos Gaussiano, Mediana y Simple.

La Interfaz Matrix se define como se trata la imagen, cuyos índices de filas y columnas identifican un punto en la imagen y el correspondiente elemento de matriz identifica el valor de gris o RGB en ese punto.

La clase Preprocessor, recibe la imagen digital capturada, luego convierte esa imagen en una matriz, evalúa esa matriz y la convierte en escala de grises; en función de esa evaluación filtra la imagen aplicando un determinado filtro.



# Diagrama de Clases del paquete Segmentation

Este diagrama describe todas las entidades para la segmentación de las imágenes preprocesadas.

Consiste en un descriptor de objetos que posee herencia hacia cualquier tipo de objeto describible que puede ser encontrado en una imagen preprocesada.

A su vez consta con una instancia implementada de descriptor de rostros que permite encontrar el rostro de personas en una determinada imagen (capture).

Cada descriptor se comunica con su propio HaarCascadeDetection. En el caso del rostro de las personas se comunica con el HaarCascadeFaceDetection. Estos son archivos .XML que se encuentran almacenados en el sistema de archivos en donde los mismos se encargar de filtrar únicamente los rostros de personas encontrados.



# Diagrama de Clase del paquete Skeleton

Este diagrama describe, el comportamiento de la detección, seguimiento de movimientos, posturas, gestos de una persona. Se destacan tres clases fundamentales: Skeleton – Joint – SkeletonStream.

La clase Skeleton describe a la persona. Posee un Id(TrackingId) por que el sensor Kinect puede detectar hasta 6 personas; dos personas en estado Tracked y las restantes en estado PositionOnly. Cuando definimos el estado Tracked, es decir detectado, puede seguir todos sus movimientos de la persona, reconociendo las articulaciones del cuerpo. Por otra parte existe estado es PositionOnly, que reconoce que hay una persona pero no puede hacer un seguimiento de sus movimientos por que ya hay dos persona que están siendo seguidas. Por último tenemos un estado NotTracked, que no detecta la presencia de una persona.

Skeleton posee una colección de la clase Joint(articulaciones del cuerpo), cada Joint posee un estado, con un tipo(cabeza,mano,pie,etc) y una posición x,y donde se encuentra , así mismo el Skeleton también tiene una posición en el espacio z. En función de esta información se realizan los algoritmos de detección de movimiento, posturas y gestos

La clase SkeletonStream recupera desde la Kinect, los datos de las personas en flujo de datos, posee un array de Skeleton que es de tamaño constante, por el hecho que detecta hasta 6 personas.



# Diagrama de clases del paquete Web

Este diagrama describe todas las entidades para el manejo de las publicidades (adverts).

Los adverts (publicidades) forman parte de una campaña (campaing). En una campaña pueden tener más de una publicidad que pertenece a un cliente además una campaña se puede ejecutar en más de un puesto publicitario.

Los adverts pueden ser los siguientes: video, catálogo y juego interactivo. Pueden añadirse más ya que existe una un mecanismo de herencia que provee dicho servicio.

La clase AdvertCampaign representa a la campaña, posee un estado tiene asociada corresponde a un Customer (cliente) con su respectivo Invoice (factura) y Estimate(cuánto dura la campaña).

La clase AdvertCampaignDetail posee una publicidad (Advert), cuando se emite y finaliza esa publicidad (tiempos de la pauta) y el precio de la publicidad.

La clase Advert despcrita anteriormente en el paquete Interaction Package es la publicidad que puede ser un Catalog(catálogo), GameMemory(juego interactivo),video. La clase Advert tiene definido una fecha inicio, fecha fin, fecha actualización y a que cliente pertenece.

La clase CatalogDetail posee un CommercialProducto (producto), continene la información que muestra un código QR y el orden que aparece en el Catálogo.

La clase GameDetail posee un CommercialProducto (producto) y muestra un de código QR y el descuento que se aplica en caso de ganar.

La clase CommercialProduct tiene definido la descripción del producto, la dirección donde está alojada la imagen en el servidor (urlPath), el nombre del mismo, los likes(son los me gusta que seleccionan las personas), views(son cuantas veces fue visto el producto) y un tipo(CommercialProductType).

Los AdvertHost son los puesto publicitarios donde se van emitir las publicidades, cada advert host tiene un id único asignado a cada Kinect.



# Diagrama de clases del paquete Monitoring

Este diagrama describe todas las entidades para el manejo flujos de personas.

Para ello emplea un servicio de monitoreo que se encarga de contar las cantidad de persona que están presente delante del sensor Kinect de un determinado puesto. Luego realiza un promedio del flujo de persona por que puede ser que las persona que detectó el sensor Kinect sean las mismas y se comunica con el modulo web para escribir sobre la base de dato las estadísticas de las personas circulantes.



# Diagrama de clases del paquete Catalog

Este diagrama describe todas las entidades para la interacción del catálogo.

Implementa las clases del paquete Interation y Web.

La clase Advert comporta polimórficamente para soportar la estructura del catálogo, se diferencia del resto de las publicidades por que se le agrega una clase llamada SortCatalog al Advert.

En SortCatalog definen 4 métodos:

* sortCustomerDefined: Ordena el Catálogo en base al orden que definió el cliente.
* sortMinPonderation: Ordena el Catalogo por ponderación de menor a mayor.
* sortMaxPonderation: Ordena el Catalogo por ponderación de mayor a menor.
* sortShuffle: Ordena el Catalogo aleatoriamente

La ponderación es igual a la proporción de a los Likes (“me gusta”) sobre la cantidad de vistas.

Para determinar la interacción del catalogo con la persona se implementa la clase CatalogDetailsInteractions, esta clase determina si una persona le dio me gusta y si vio los productos, determinando en qué momento se realizó la interacción y el tiempo que duró.



# Diagrama de clases del paquete GameMemory

Este diagrama describe todas las entidades para la interacción del juego de la memoria. Consiste en seleccionar dos productos iguales utilizando tres vidas (intentos de juego) de una grilla de productos ocultos, es básicamente un tipo de juego puzzle memory.

Implementa las clases del paquete Interacción y Web.

Se implementa la clase Advert del paquete Web, se le agrega una colección de la clase GameDetails, esta clase se define el producto a comparar, la información del código QR y descuento en caso de ganar.

Para determinar la interacción del juego con la persona se implementa la clase GameDetailInteractions, esta clase determina si una persona ganó o no, el tiempo de interacción y en qué momento realizó la interacción. Si una persona gana determina con que producto ganó y me muestra un código de QR al ganador.

Se implementa también la clase AdvertVideo, la cual define como se reproduce el video publicitario y la ruta donde se encuentra el video.



# Diagrama de estado de KinectStateMachine

En este diagrama se describe el funcionamiento de la máquina de estado del sensor Kinect, tratando todos los estados que puede tomar.

La Maquina de estado del sensor Kinect se crea tomando el estado Initilizating, que es cuando, el sensor Kinect se conecta al host y se prepara para pasar el estado Connected, creando y habilitando los procesos que se necesiten en función del objetivo que tenga que cumplir el sensor Kinect, por ejemplo realizar el seguimiento de una persona, se habilitan determinados procesos. En el estado Initilizating además de pasar a Connected, puede pasar a un estado NotPowered en el caso de que no reciba o se perdió la corriente eléctrica del sensor Kinect.

Por otra parte puede pasar al estado Error en caso que se produzca error porque no se pueda iniciar algún proceso por que falten parámetros no inicializado o por que la Kinect no es original. Además existe el estado Disconnected en caso de que se pierda la conexión de Datos(USB) o Se evaluó que por ciertos parámetros o procesos no se puede conectar.

En el Estado NotPowered, el sensor Kinect no posee energía eléctrica se informa al usuario para que restablezca la conexión eléctrica para luego pasar al estado Initilizating, y si persiste la falta de no poder restablecerse la conexión eléctrica se pasa al estado Error informando de la situación.

En el Estado Connected, ya están habilitados los servicios para ser utilizados, el sensor Kinect se encuentra listo para transmitir Stream(flujo de datos) , en este estado puede también pasar al estado NotPowered, se puede volver al estado Initilizating en caso de querer cambiar la configuración y habilitar otros servicios. En caso de haber fallas con algunos de los componentes del sensor(cámaras, motor, acelerómetro, micrófonos) pasa al estado Error y se pasa al estado Disconnected en caso de querer terminar la operación del sensor.

En el estado Disconnected en este estado el sensor se encuentra desconectado por diferentes motivos (falla en los componentes, sin señal de datos, se terminó la operación del sensor Kinect), se puede pasar al estado Initilizating en caso de querer iniciar determinados servicios para realizar un correspondiente objetivo. En caso de haber fallas se pasa al estado error en caso de querer informar algún error de por qué se llegó al estado Disconnected.

En el estado Error, se informa todos los errores al usuario y luego se lo guarda a todos los errores indicando el tipo de error con su identificación y prioridad.



# Diagrama de Estado de AdvertStateMachine

En este diagrama se describe el funcionamiento de una publicidad detallando por todos los estados que pasa.

En el estado Preparing, aquí el host (puesto publicitario) solicita a la base de datos que la información sobre la publicidad que va emitir, luego conociendo que imágenes y videos va necesitar , le solicita al servidor HTTP mediante una petición las imágenes y los videos para guardarlo y cambia al Estado Disclaiming.

En el estado Disclaiming, el host detecta la presencia de una persona y informa que se va ejecutar una publicidad interactiva en la cual va extraer datos de esa interacción por cuestiones legales, si acepta se cambia al estado Snaping.

En el estado Snaping se puede aplicar dos estrategias en función del patrón Strategy: Algoritmo interactivo para anuncios o algoritmo de monitoreo para rastrear el movimiento de las personas.

En principio se encuentra en modo monitoreo se va ejecutar de 1 a n veces siempre que la persona se encuentra mayor a los 2 metros pero si se encuentra menor a los dos metros se ejecuta el algoritmo interactivo. Si se encuentra en el modo monitoreo se ejecuta una publicidad convencional y extrae datos de las personas pasa al estado Saving.

Si se encuentra en el modo interactivo se ejecuta la publicidad interactiva pasa al estado Running. En el estado Running Se ejecuta la publicidad interactiva (puede ser el Catálogo), se captura todos los datos que se va generando en la interacción de la persona con la publicidad luego pasa al estado Saving para guardar esa información.

En el estado Saving si se llega por el modo monitoreo se procesan las imágenes digitales y se guarda en la base de datos y también si se llega por el modo interactivo se guarda la información generada en la base de datos. Se puede volver al estado Saving si se encuentra en el modo monitoreo para seguir extrayendo datos de las personas si no se puede terminar la interacción pasando al estado Closing.

En el estado Closing se terminó la publicidad y puede volver a iniciarse el ciclo si todo está en condiciones cambiando al estado Preparing o sino terminar el ciclo e informar la situación.



# Diagrama de Estado de Skeleton

Este diagrama describe el comportamiento de la captura de un Skeleton(persona) para poder comenzar el anuncio.

En el estado Tracked se detecta la persona para poder realizar el seguimiento de movimientos, posturas, gestos y se inicia el anuncio contando el tiempo de inicio, cuando el sensor pierde el seguimiento de la persona se pasa al estado Inferred.

En el Estado Inferred se perdió momentáneamente el seguimiento de una persona, busca a la persona para pasar al estado Tracked en un determinado tiempo de delay; pasado el tiempo se pasa al estado NotTracked.

En el estado NotTracked, se perdió la persona, no hay seguimiento de persona se determina el tiempo que estuvo en estado Tracked en función del tiempo de finalización, se guardan las estadísticas.



# Diagrama de Estado de Catálogo

Este diagrama describe el comportamiento interacción del catálogo.

En el estado SetupResources se setea todos los recursos descargados que se mostraran en el catálogo. Se descargan las imágenes de todos los productos de los catálogos requeridos en un determinado AdvertHost.

En el Estado StaticFlow es cuando no detecta persona muestra publicidades estáticas a la espera que una persona interactúe con el catálogo. Cuando detecta que una persona quiere interactuar pasa al estado DynamicFlow, comienza mostrando la selección del catálogo a ser vistos.

En ViewCatalog es cuando selección el catalogo que quiere ver, reproduce el catalogo mostrando los producto con los que va interactuar la persona. Puede cambiar al estado PlayingAdvertVideo si la persona se fue del puesto publicitario (AdvertHost).

En el estado ViewtItem es cuando seleccionó el producto que quiere ver. Puede pasar al estado ViewCatalog si selecciona volver a ver el catalogo. Puede pasar al estado ValueItem si le dio me gusta a la imagen. Puede pasar al estado PlayingAdvertVideo si la persona se fue del puesto.

En el estado ValueItem se produce cuando una persona le da me gusta al producto, pero puede pasar que una persona quiera deseleccionar el me gusta.

En el estado PlayingAdvertVideo es cuando la persona deja de interactuar con el catalogo, comienza a reproducir videos publicitarios.

En el estado Error ocurre si se producen error con el sensor Kinect reportando información a la base de datos.



# Diagrama de Estado de Juego Game Memory

Este diagrama describe el comportamiento interacción del juego Game Memory

En el estado Preparing se ordenan los productos a interactuar aleatoriamente y se carga el video publicitario a reproducir.

En el Estado Ready es cuando el juego está listo para ejecutarse. Pasa al estado WaitFigure2 cuando selecciona un producto oculto, lo muestra y espera que se seleccione otro producto, puede pasar al estado Checked si selecciono el segundo producto. Puede pasar al estado PlayingAdvertVideo si la persona se fue del puesto.

En el estado Checked es selecciona el segundo producto, verifica si es igual al primero que selecciono, si son iguales pasa al estado Win si no pasa al estado Ready si le quedan (intentos) vida para volver a intentar de elegir dos productos iguales. Si no le quedan vidas (intentos los cuales son 3) pasa al estado GameOver.

En el estado Win es cuando se eligen dos productos iguales en una vida, se detiene el video publicitario, muestra el descuento que gano con ese producto y un código QR.

En el estado PlayingAdvertVideo es cuando la persona deja de interactuar con el juego, comienza a reproducir videos publicitarios.

En el estado GameOver es cuando la persona utilizo todas las vidas del juego.



# Diagrama de Despliegue de OpticalMarketing

En este diagrama se describe el despliegue del sistema OpticalMarketing.

Los AdvertHost son los clientes del puesto publicitario y pueden existir de uno a N.

El puesto se compone por un CPU y una pantalla de 32 "o 42". El mismo tiene conectado un dispositivo Sensor Kinect y la correspondiente aplicación cliente con un framework NET 4.0 y Microsoft Kinect SDK para concretar la interacción con el mismo.

Se emiten anuncios con interacciones e información, se genera información y feedback la guarda en el DatabaseServer. También solicita videos o imágenes al OpticalMarketingServer necesarios para emitir los anuncios. La información almacenada en el OpticalMarketingServe será consultada por nuestro cliente en los host CustomerClients.

OpticalMarketingServer, es un servidor centralizado que soporta aplicaciones distribuidas, entre ellas un servidor web y un servidor de base de datos.

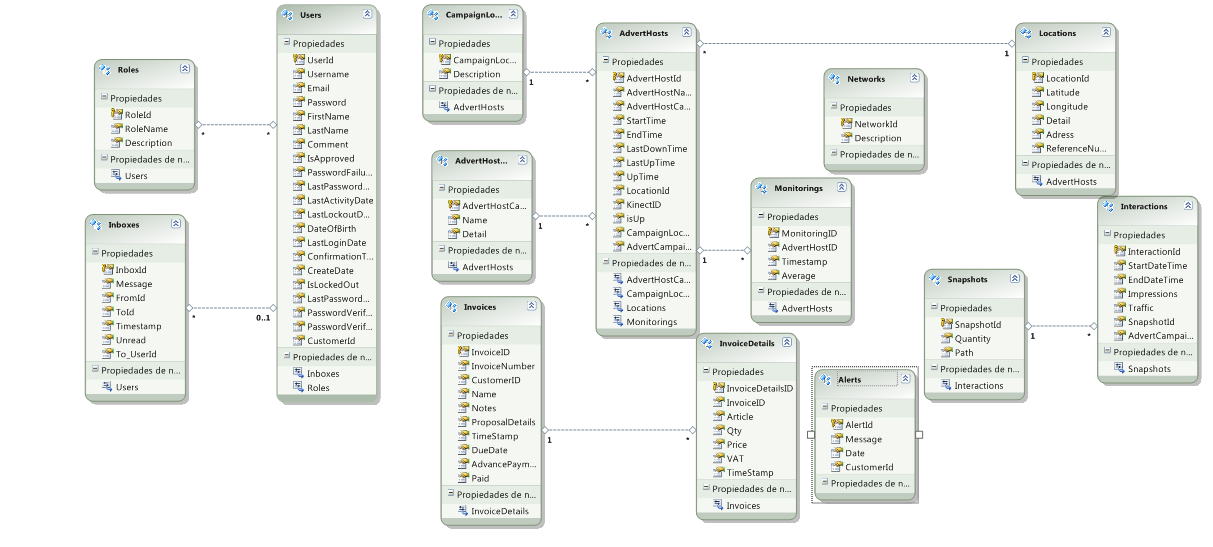
WebServer: un servidor web que se encuentra en OpticalMarketingServer que opera para recibir consultas y transacciones usando HTTP, HTTPS y WS. Implementa la instancia de Internet Information Server (IIS).

DataBase Server: Base de datos centralizada de OpticalMarketing. Esta base implementa data mining bajo la plataforma Microsoft SQL Server. Está ubicada en OpticalMarketingServer.

CustomerClient: Es un cliente que quiere contratar a través de un navegador Web, servicios de OpticalMarketing. El cliente puede administrar sus anuncios, ver gráficos e información de los anuncios emitidos en los AdvertHost, pagar los servicios brindados por OpticalMarketing a través de los servicios de PayPal.



# Diagrama Entidad Relación de bases de datos relacional

El equipo utiliza EntityFramework que actualiza constantemente la BD, pero este se basa en el diagrama de clases original para realizar el mapeo correspondiente.

