**UNO BLUETOOTH GAME**

López González Arturo

Marquez Quezada Gabriela

Muñoz Mitre Luis Ángel

Sampayo García José Luis

Susarrey Alejandro

Valle Héctor Alfonso

Maestría en Ciencias Computacionales

Programación y Arquitectura de Sistemas Distribuidos

Gustavo Torres Blanco

# **HISTORIAL DE CAMBIOS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Fecha de Revisión** | **Modificado por** | **Revisado por** | **Modificaciones** | **Comentarios** |
| 15-04-2015 | G. Marquez | G. Marquez | Creación de base del documento |  |
| 16-04-2015 | A. López | G. Marquez | Secciones 3.5.1 y 5.2 |  |
| 18-04-2015 | G. Marquez | G. Marquez | Capítulo 3 y Sección 6.1.1 | Excepto la sección 3.5.1 |
| 20-04-2015 | G. Marquez | G. Marquez | Capítulo 4 y 5 | Excepto la sección 5.2 |
| 25-04-2015 | G. Marquez | G. Marquez | Capítulos 6 y 7 | Excepto la sección 6.1.1 |
| 27-04-2015 | A. López | G. Marquez | Capítulo 8 |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

1. **CONTENIDO**

[**1.** **HISTORIAL DE CAMBIOS** 2](#_Toc449561711)

[**2.** **CONTENIDO** 3](#_Toc449561712)

[**3.** **INTRODUCCIÓN** 5](#_Toc449561713)

[3.1. Propósito 5](#_Toc449561714)

[3.2. Alcance 5](#_Toc449561715)

[3.3. Términos y Abreviaciones 5](#_Toc449561716)

[3.4. Referencias 5](#_Toc449561717)

[3.5. Descripción del proyecto 6](#_Toc449561718)

[3.5.1. Descripción de la aplicación 6](#_Toc449561719)

[3.5.2. Objetivos 6](#_Toc449561720)

[3.5.3. Justificación 6](#_Toc449561721)

[3.5.4. Limitaciones del proyecto 6](#_Toc449561722)

[**4.** **DISEÑO DE INVESTIGACION** 7](#_Toc449561723)

[4.1. Diseño Conceptual del Producto 7](#_Toc449561724)

[4.2. Arquitectura Modular 7](#_Toc449561725)

[**5.** **PLANEACIÓN** 8](#_Toc449561726)

[5.1. Planeación del Proyecto 8](#_Toc449561727)

[5.1.1. Recursos necesarios para realizar el Proyecto 8](#_Toc449561728)

[5.1.2. Plan de Trabajo 8](#_Toc449561729)

[5.2. Especificaciones Técnicas 10](#_Toc449561730)

[5.2.1. Sistema Distribuido 10](#_Toc449561731)

[5.2.2. Protocolo de Comunicación Bluetooth 11](#_Toc449561732)

[5.2.3. UNO 13](#_Toc449561733)

[5.3. Modelo Conceptual 14](#_Toc449561734)

[**6.** **ESPECIFICACIONES FUNCIONALES** 15](#_Toc449561735)

[6.1. Arquitectura 15](#_Toc449561736)

[6.1.1. Modelo de roles para la comunicación 15](#_Toc449561737)

[6.1.2. Modelo de Comunicación Publish-Subscribe 15](#_Toc449561738)

[6.1.3. Picoredes y Redes Dispersas WPAN Bluetooth. 16](#_Toc449561739)

[6.1.4. Diagrama de interacciones 17](#_Toc449561740)

[**7.** **CASOS DE USO** 18](#_Toc449561741)

[7.1. Comenzar juego 18](#_Toc449561742)

[7.2. Unirse a juego 18](#_Toc449561743)

[7.3. Realizar jugada 19](#_Toc449561744)

[7.3.1. Carta disponible 19](#_Toc449561745)

[7.3.2. Pasar turno 20](#_Toc449561746)

[7.3.3. Tomar carta de la pila 20](#_Toc449561747)

[7.4. Conocer jugada del jugador en turno 21](#_Toc449561748)

[7.5. Cambio de turno 22](#_Toc449561749)

[7.6. Ausencia del jugador en turno 22](#_Toc449561750)

[7.7. Dar de baja a un jugador 23](#_Toc449561751)

[7.8. Jugador ha ganado 23](#_Toc449561752)

[7.9. Fin de juego 24](#_Toc449561753)

[**8.** **ESPECIFICACIÓN DE CÓDIGO** 24](#_Toc449561754)

[8.1. MainActivity (Conexión handler) 24](#_Toc449561755)

[8.2. FuncionesJava 24](#_Toc449561756)

[8.3. CommunicationModuleA 25](#_Toc449561757)

[8.4. SecondActivity (Juego) 25](#_Toc449561758)

# **INTRODUCCIÓN**

## Propósito

Este documento tiene como propósito dar a conocer los aspectos principales que conforman el proyecto, los factores que nos llevaron a su realización, así como los fundamentos teóricos, planeación y desarrollo del mismo.

## Alcance

Este documento contiene el desarrollo del Proyecto “Uno Bluetooth Game”, la información relacionada a la descripción y objetivos del proyecto, así como las especificaciones funcionales.

## Términos y Abreviaciones

**API-** Por sus siglas en inglés Application Programming Interface, es el conjunto de subrutinas, funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

**Dirección MAC-** Por sussiglas en inglés “media access control” o en español "control de acceso al medio", es un identificador de 48 bits (6 bloques hexadecimales) que corresponde de forma única a una tarjeta o dispositivo de red. Se conoce también como dirección física, y es única para cada dispositivo

**TCP-** Transmission Control Protocol o Protocolo de Control de Transmisión

**UML-** Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified Modeling Language). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema

## Referencias

* http://www.hpcs.cs.tsukuba.ac.jp/~tatebe/lecture/h23/dsys/dsd-tutorial.html
* https://www.bluetooth.com/what-is-bluetooth-technology/bluetooth
* https://en.wikipedia.org/wiki/Uno\_(card\_game)
* http://developer.android.com/intl/es/guide/topics/connectivity/bluetooth.html
* El estándar Bluetooth IEEE802.15.1: http://catarina.udlap.mx

## Descripción del proyecto

### Descripción de la aplicación

Se desarrollará un juego de UNO sobre la plataforma de Android y este será multijugador uniendo diferentes dispositivos mediante la comunicación Bluetooth. Por medio del protocolo de comunicación se enviaran mensajes empaquetados (marshalling) y cada dispositivo hará la decodificación del paquete (un-marshalling).

La aplicación será un sistema distribuido ya que no tendrá un servidor centralizado que sea el dueño de la partida, sino que se repartirán los recursos para des-centralizarlo y de esta manera cualquiera de los dispositivos podrá tener el estado del juego sin necesidad de ir a un sitio remoto a obtenerlo.

### Objetivos

* Diseñar un sistema distribuido basado en el tradicional juego de cartas UNO.
* Utilizar el protocolo de comunicaciones Bluetooth en un juego multijugador para más de 2 dispositivos.
* Asegurar el correcto funcionamiento de la aplicación al cumplir los 8 Challenges de los Sistemas Distribuidos (Heterogeneidad, Apertura, Seguridad, Escalabilidad, Manejo de Fallas, Concurrencia, Transparencia, Calidad de Servicio).

### Justificación

Ofrecer a los fanáticos del juego UNO una opción más para divertirse como es un juego multijugador no convencional ya que a diferencia de lo existente en el mercado utiliza el protocolo de comunicación Bluetooth en lugar de las redes de telefonía celular (3G, 4G, 4G LTE) o Wi-Fi.

### Limitaciones del proyecto

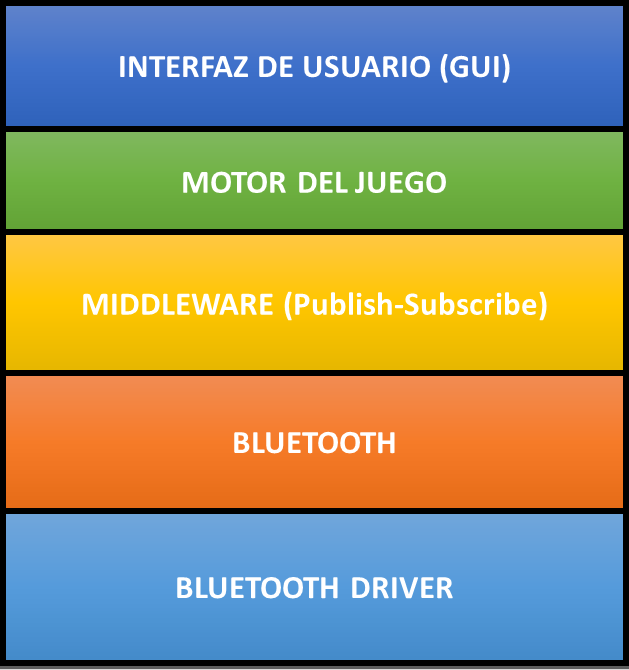
* Bluetooth es una red de corto alcance y solo permite tener un máximo de 7 dispositivos en red.
* Corto tiempo de desarrollo de la aplicación.
* Disponibilidad de los recursos.

# **DISEÑO DE INVESTIGACION**

## Diseño Conceptual del Producto



## Arquitectura Modular



# **PLANEACIÓN**

## Planeación del Proyecto

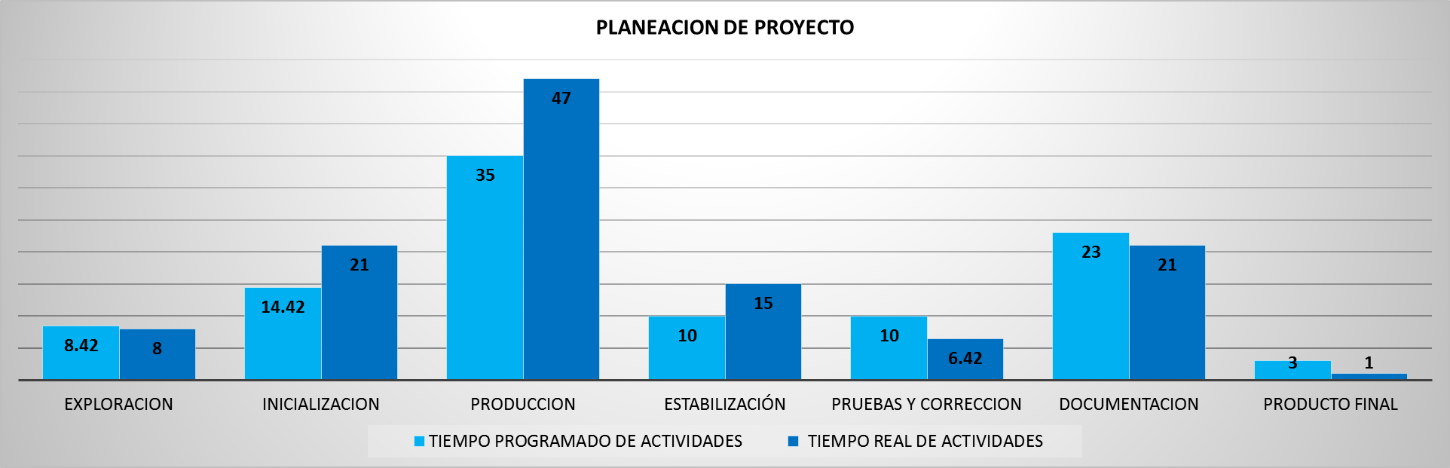
* + 1. Recursos necesarios para realizar el Proyecto
* *Materiales*
  + Computadora de escritorio o laptop para desarrollo de la aplicación
  + Dispositivo con Sistema Operativo Android y conectividad Bluetooth
  + Entorno de Desarrollo Integrado para el desarrollo de aplicaciones móviles (Android Studio)
  + Software para planeación y organización de actividades (Microsoft Visio, Proyect, Excel o similar)
  + Software para desarrollo y documentación de pruebas de la aplicación (IBM jazz.net)
  + Software para realización de diagramas orientados a redes y desarrollo de software (Microsoft Visio, StarUML)
  + Licencias de uso de software (en caso de requerirse)
* *Humanos*

Personal con conocimiento sobre:

* + Programación Orientada a Objetos
  + Programación de aplicaciones para dispositivos móviles (Android SDK)
  + Programación de Juegos (Interfaz de Usuario, Animaciones y Lógica de Juego)
  + Manejo de Android Studio
  + Realización de diagramas utilizando UML
  + Diseño y Funcionamiento de los Sistemas Distribuidos
  + Protocolo de Comunicación Bluetooth
  + Gestión de Proyectos de Software
    1. Plan de Trabajo



* Grafico Comparativo de Tiempo Planeado vs Real



* Distribución de Etapas de Proyecto
* Plan de Trabajo Real a detalle:



## Especificaciones Técnicas

## Sistema Distribuido

* Definición:

Un sistema distribuido: es una aplicación que ejecuta una colección de protocolos para coordinar las acciones de múltiples procesos en una red, de tal manera que todos los componentes cooperan juntos para realizar una sola o un pequeño conjunto de tareas relacionadas.

* Elementos de un Sistema Distribuido:
  + Un programa: es el código que escriba.
  + Un proceso: es lo que obtienes cuando lo ejecuta.
  + Un mensaje: se utiliza para la comunicación entre procesos.
  + Un paquete: es un fragmento de un mensaje que pueda viajar en un alambre.
  + Un protocolo: es una descripción formal de formatos de mensaje y las normas que deben seguir dos procesos con el fin de intercambiar esos mensajes.
  + Una red: es la infraestructura que conecta ordenadores, estaciones de trabajo, terminales, servidores, etc.
  + Un componente: puede ser un proceso o de cualquier pieza de hardware necesario para ejecutar un proceso, de comunicaciones de soporte entre los procesos, los datos de tienda, etc.
* Características ideales de un Sistema Distribuido:
  + Tolerancia a fallos: puede recuperarse de fallas de los componentes sin realizar acciones incorrectas.
  + Alta disponibilidad: Puede restaurar las operaciones, que le permite reanudar la prestación de servicios, incluso cuando algunos componentes han fracasado.
  + Recuperables: Los componentes defectuosos pueden reiniciar y volver a unirse a sí mismos al sistema, después de que la causa de la falla ha sido reparado.
  + Consistente: El sistema puede coordinar la acción de múltiples componentes a menudo en presencia de concurrencia y el fracaso. Esto subraya la capacidad de un sistema distribuido de actuar como un sistema no distribuido.
  + Escalable: Puede funcionar correctamente incluso como algunos aspectos del sistema se expanden a un tamaño mayor. Por ejemplo, podríamos aumentar el tamaño de la red en la que el sistema está funcionando. Esto aumenta la frecuencia de las interrupciones de la red y podría degradar un sistema de "no escalable". Del mismo modo, podríamos aumentar el número de usuarios o servidores, o la carga total en el sistema. En un sistema escalable, esto no debería tener un efecto significativo.
  + Rendimiento predecible: La capacidad de proporcionar la respuesta deseada en el momento oportuno.
  + Seguro: El sistema autentique el acceso a datos y servicios.

## Protocolo de Comunicación Bluetooth

Creado en 1994, la tecnología Bluetooth® se concibió como una alternativa inalámbrica para cables de datos mediante el intercambio de datos a través de las transmisiones de radio. El nombre de Bluetooth del rey danés, Harald Blåtand (Harold Bluetooth en inglés). Según la historia, el rey Blåtand ayudó a unir las facciones enfrentadas en partes de lo que hoy son Noruega, Suecia y Dinamarca; Del mismo modo, la tecnología Bluetooth fue creado como un estándar abierto para permitir la conectividad y la colaboración entre los productos y las industrias dispares.

Una de las aplicaciones más populares para Bluetooth históricamente ha sido de audio-auriculares inalámbricos y conectividad manos libres en los coches de altavoces inalámbricos y auriculares que transmiten la música desde su teléfono o tableta. Este sistema utiliza una versión de Bluetooth llama BR / EDR (bit rate/enhanced data rate) que está optimizado para el envío de un flujo constante de datos de alta calidad (música) economizando energía.

BR / EDR y Bluetooth con bajo consumo de energía son fundamentalmente diferentes. Bluetooth con la funcionalidad de bajo consumo de energía se basa en un completamente nuevo marco de desarrollo utilizando atributos genéricos, o GATT. GATT es extremadamente flexible desde la perspectiva de un desarrollador y se puede utilizar para casi cualquier escenario. Como resultado, Bluetooth no sólo conecta dispositivos entre sí de una manera eficiente ultra-poder, pero también se conecta directamente a los dispositivos para aplicaciones en el smartphone, PC o tableta. Es el bajo consumo de energía y las características del GATT que están en el corazón del actual auge de la Internet de las cosas (IoT).

La plataforma Android incluye soporte para la pila de red Bluetooth, que permite a un dispositivo para intercambiar datos de forma inalámbrica con otros dispositivos Bluetooth. El marco de aplicación proporciona acceso a la funcionalidad Bluetooth a través de las API de Android Bluetooth. Estas API permiten aplicaciones de forma inalámbrica conectarse a otros dispositivos Bluetooth, punto a punto y multipunto que permite funciones inalámbricas.

El uso de las API de Bluetooth, una aplicación para Android puede realizar lo siguiente:

* Buscar otros dispositivos Bluetooth
* Consultar el adaptador Bluetooth local para los dispositivos Bluetooth vinculados
* Establecer canales RFCOMM
* Conectarse a otros dispositivos a través de la detección de servicios
* Transferir datos hacia y desde otros dispositivos
* Gestión de múltiples conexiones
  + - 1. *Bluetooth clásico*

Bluetooth Clásico es la elección correcta para más operaciones de mayor consumo como el streaming y la comunicación entre los dispositivos Android. Para los dispositivos Bluetooth con bajos requerimientos de energía, Android 4.3 (API Nivel 18) introduce soporte para la API de Bluetooth de baja energía.

Las clases e interfaces que se necesitan para crear conexiones de Bluetooth son:

* BluetoothAdapter: Representa el adaptador Bluetooth local (radio Bluetooth), es el punto de entrada para toda la interacción Bluetooth. Con el uso de este, se pueden descubrir otros dispositivos Bluetooth, consultar la lista de dispositivos enlazados (emparejadas), una instancia de un BluetoothDevice mediante una dirección MAC conocida, y crear un BluetoothServerSocket para escuchar las comunicaciones de otros dispositivos.
* BluetoothDevice: Representa un dispositivo Bluetooth remoto. Esta opción se utiliza para solicitar una conexión con un dispositivo remoto a través de un BluetoothSocket o consulta de información sobre el dispositivo, tales como su nombre, dirección, clase y estado de enlace.
* BluetoothSocket: Representa la interfaz para una toma de Bluetooth (similar a un Socket TCP). Este es el punto de conexión que permite una aplicación para intercambiar datos con otro dispositivo Bluetooth a través de InputStream y OutputStream.
* BluetoothServerSocket: Representa un servidor socket abierto que escucha las solicitudes entrantes (similar a un ServerSocket TCP). Con el fin de conectar dos dispositivos Android, un dispositivo debe abrir un socket servidor con esta clase. Cuando un dispositivo Bluetooth remoto realiza una solicitud de conexión al este dispositivo, el BluetoothServerSocket devolverá un BluetoothSocket conectado cuando se acepta la conexión.
* BluetoothClass: Describe las características generales y capacidades de un dispositivo Bluetooth. Se trata de un conjunto de sólo lectura de las propiedades que definen las clases de dispositivo mayor y menor del dispositivo y los servicios. Sin embargo, esto no describe de forma fiable todos los perfiles de servicios Bluetooth compatibles con el dispositivo, pero es útil como una sugerencia para el tipo de dispositivo.
* BluetoothProfile: Una interfaz que representa un perfil de Bluetooth. Un perfil Bluetooth es una especificación de la interfaz inalámbrica para la comunicación basada en Bluetooth entre dispositivos. Un ejemplo es el perfil de manos libres. Para mayor discusión de los perfiles, consulte Funcionamiento con perfiles.

## UNO

Uno es un juego de cartas americano que se juega con una baraja especialmente impresa. El juego fue desarrollado originalmente en 1971 por Merle Robbins en Reading, Ohio, un suburbio de Cincinnati. Ha sido una marca Mattel desde 1992. Los principios generales del juego es poner las cartas por orden de color o de número siguiendo la secuencia.

Para iniciar, siete cartas se reparten a cada jugador, y la primera carta de la baraja se voltea y dejar de lado para comenzar la pila de comer. El jugador a la izquierda del repartidor juega primero, a menos que la primera carta en la pila de descarte es una acción o comodín. En el turno de un jugador, él / ella debe hacer una de las siguientes:

* Jugar una carta que coincida con la carta del centro en color , número o símbolo
* Jugar un comodín , o un toma cuatro
* Tomar la primera carta de la baraja

El juego continúa en sentido horario alrededor de la mesa.

## Modelo Conceptual



# **ESPECIFICACIONES FUNCIONALES**

## Arquitectura

### Modelo de roles para la comunicación



### Modelo de Comunicación: Sistemas de mensajería

Los sistemas de mensajes, constituyen un paradigma basado en una capa intermedia (middleware) que gestiona el paso de mensajes entre componentes. Por ello, se suele hablar de una capa intermedia del paradigma de paso de mensajes. El nombre más conocido de este paradigma es “Message-Oriented Middleware (MOM)”. Existen dos modelos de implementación de este paradigma: punto a punto (point to point) y publicación/suscripción (publish/subscribe).

En el modelo punto a punto, el sistema de mensaje emplea colas que permiten desacoplar el emisor del receptor. Cuando un componente desea enviar un mensaje a otro, deposite el mensaje en la cola del receptor (lo que implica conocerlo) y se desentiende del mensaje.

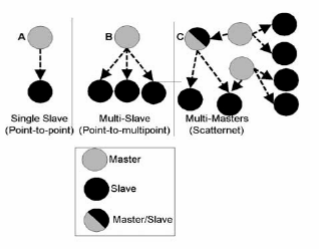
En el modelo de publicación/suscripción, cada mensaje es asociado a un evento específico, de manera que los componentes que estén interesados en recibir los mensajes asociados al evento, se suscriben al mismo. El sistema de mensajes es el responsable de la gestión completa de los mensajes.

• “peer to peer”. Con algunos conflictos en la traducción, suele hablarse de modelo entre pares, o entre iguales. En este paradigma las funcionalidades de cada componente son las mismas. En este caso las operaciones que puede hacer cada componente son: solicitud y respuesta. Protocolos como VoIP, usado por skype, están basados en este paradigma. Generalmente se suele combinar este paradigma con el de cliente servidor para establecer las conexiones.

• “Publish-subscriber”. Este modelo soporta las comunicaciones de un emisor a muchos receptores con soporte a la redundancia tanto en las publicaciones, como en los receptores que se suscriben a las fuentes de información.

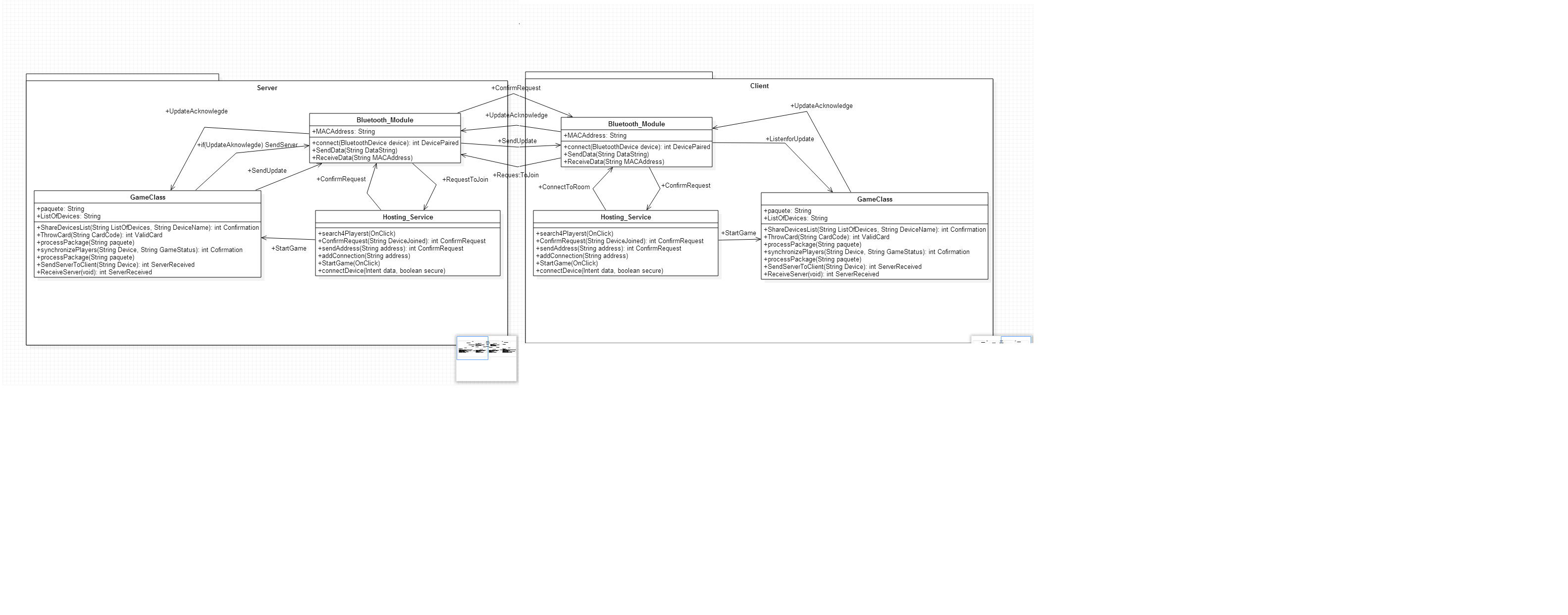
### Picoredes y Redes Dispersas WPAN Bluetooth.

Una picored es una WPAN formada por dispositivos Bluetooth que sirven como maestros en la picored y uno o más dispositivos Bluetooth que sirven de esclavos. Un canal de frequency-hopping basado en la dirección del maestro define cada picored. Todos los elementos participantes en una comunicación dentro de una picored dada son sincronizados al canal de frequency-hopping, utilizando el reloj del aparato maestro de la picored. Los dispositivos esclavos solo se comunican con el maestro en una estructura punto a punto bajo el control del maestro. Las transmisiones del maestro deben de ser ya sea de la forma punto a punto o bien punto a multipunto. Cada lugar determina que si algunos elementos deben ser siempre maestros o esclavos. Sin embargo, este estándar no distingue entre elementos que son permanentemente maestros o esclavos. Durante una sección de comunicación un elemento esclavo puede comportarse como maestro en un momento dado y viceversa.

Una red dispersa (scatternet) es una colección de varias picoredes en operación que se traslapan en tiempo y en espacio. Un aparato Bluetooth puede participar en múltiples picoredes al mismo tiempo, por lo que existe la posibilidad de que la información fluya más allá de las fronteras de una picored. Un aparato en una scatternet puede ser esclava en varias picoredes, pero puede ser maestra en solo una de ellas. La figura muestra las diferentes formas en que los dispositivos Bluetooth se pueden interconectar para formar varios sistemas de comunicación

Varios formatos en una picored. (a) Operación con un solo esclavo. (b) Operación multiesclavo; (c) Operación tipo scatternet.

### Diagrama de interacciones



# **CASOS DE USO**

* 1. Comenzar juego



* 1. Unirse a juego



* 1. Realizar jugada



* + 1. Carta disponible



* + 1. Pasar turno



* + 1. Tomar carta de la pila



* 1. Conocer jugada del jugador en turno



* 1. Cambio de turno



* 1. Ausencia del jugador en turno



* 1. Dar de baja a un jugador



* 1. Jugador ha ganado



* 1. Fin de juego



# **ESPECIFICACIÓN DE CÓDIGO**

## MainActivity (Conexión handler)

* void giveItSomeTime(void);
* void stopModule(void);
* void showMessage(String message);
* private void sendMessage(String message);
* void synchronizePlayers(void);
* void processPackage(String paquete);
* void sendAddress(String address);
* void addConnection(String address);
* void search4Players(void);
* void wait4Players(void);
* void setUpCommunicationModule(void);
* void printStatus(void);

## FuncionesJava

* public void onClick(View v);
* public void onStart(void);
* public void onStop(void);
* public void onDestroy(void);
* public void onResume(void);
* public void onFinish(void);
* protected void onCreate(Bundle savedInstanceState);

## CommunicationModuleA

* private synchronized void setPlayersOnline(int playersOnline);
* public synchronized int getState(void);
* public synchronized int getPlayersOnline(void);
* public synchronized void start(void);
* public synchronized void connect(BluetoothDevice device, boolean secure);
* public synchronized void connected(BluetoothSocket socket, BluetoothDevice device, final String socketType);
* public synchronized void stop(void);
* private void connectionFailed(void);
* private void connectionLost(void);
* private class AcceptThread extends Thread {

public AcceptThread(boolean secure)

public void run(void)

public void cancel(void)

} //Manejo de en la manejo de múltiple conexiones aceptadas

* private class ConnectThread extends Thread

{

public ConnectThread(BluetoothDevice device, boolean secure)

public void run(void)

public void cancel(void)

}//Manejo para las conexiones a realizar con hilos

* private class ConnectedThread extends Thread

{

public ConnectedThread(BluetoothSocket socket, String socketType)

public void run(void)

public void write(byte[] buffer)

public void cancel(void)

}//Manejo de hilos para las conexiones ya realizadas

## SecondActivity (Juego)

* void inicio(void);
* boolean validarturno(void);
* void disponibilidaddecartas(void);
* boolean validarcarta(String s);
* void createCommunication(void);
* void tirarcarta(String s, int n);
* void AsignarCartasInicio(int p);
* void AcomodarCartasJugador’n’(void); //Existen n funciones para los distintos jugadores
* void menudecolores(void);
* void reordenarcartasjugador’n’(int id);
* void cambiarimagendelcentro(String vs);
* void cambiardireccion(void);
* void cambiarturno(void);
* void cartaAleatoria(void);
* void setUpCommunicationModule(void);
* void defineRole(String paquete);
* void giveItSomeTime(void); //Delay del temporizador
* void startCommunicationEngine(void);
* private void connectDevice(Intent data, boolean secure);
* private void sendMessage(String message);
* void notifyOthers(String message);
* void connectionLost(void)