# UNIVERSIDAD NACIONAL DE INGENIERÍA

## FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA


# SUPERVISIÓN DE PARÁMETROS DE PROCESO BASADO EN COMUNICACIÓN BLUETOOTH Y EQUIPO MÓVIL

**INFORME DE SUFICIENCIA**

PARA OPTAR EL TÍTULO PROFESIONAL DE:

**INGENIERO ELECTRÓNICO**

**PRESENTADO POR:**

**HENRY VILCHEZ ESPINOZA**

**PROMOCIÓN**
**1996 - I**

**LIMA – PERÚ**
**2012**

# SUPERVISIÓN DE PARÁMETROS DE PROCESO BASADO EN COMUNICACIÓN BLUETOOTH Y EQUIPO MÓVIL

A mis padres, hermanos, hermana, cuñadas, sobrinas y ahijado por todo el apoyo de muchas maneras en superarme cada día más. Este título es para ustedes.

## SUMARIO

Cada día más y más se requiere que el supervisor de planta se encuentre en campo verificando el correcto funcionamiento de la planta, viendo en forma directa sobre todo los equipos principales. Pero también tener la posibilidad de observar los datos de proceso desde una Interface Máquina-Humano (Human Machine Interface - HMI) que se encuentra en la sala de control. Ello asegura la continuidad del proceso y permite eliminar pérdidas en la producción por paradas de emergencias no programadas y más en las pruebas de comisionamiento por ser la primera vez que se ponen operativos los equipos principales.

Se presenta este informe planteando una solución siguiendo la tendencia actual que es el aumento de la movilidad de los empleados en las compañías [1].

El primer capítulo trata de los conceptos a utilizar durante la lectura del informe.

En el segundo capítulo se amplía los conceptos que involucra el problema planteado para su mejor entendimiento e identificar los requerimientos a cubrir tanto en hardware como software. Así se obtiene una solución con la mayor eficiencia, alcance, practicidad, compatibilidad y que origine el menor costo de inversión.

En el tercer capítulo se plantea las posibles tecnologías que permitan obtener una mejor solución.

En el último capítulo se describen las conclusiones y recomendaciones que avalan la solución planteada con el uso de un teléfono móvil en la etapa de comisionamiento.

En la última parte del informe se presentan los anexos que complementan los capítulos desarrollados incluyendo la bibliografía con el listado de todas las fuentes de información.

# ÍNDICE

# INTRODUCCIÓN

Una de las tecnologías que está alcanzando mayor énfasis en los sistemas de control es la aplicación de comunicación inalámbrica y dentro de ellos está el Bluetooth. El planteamiento de este informe es aprovechar esta tendencia para el aseguramiento del funcionamiento de la planta, pero sobre todo en las pruebas de comisionamiento de los equipos principales. Esto se logra aumentando el libre desplazamiento de los supervisores en la planta sin perder los beneficios de estar en la sala de control visualizando los datos de proceso. Así el título del informe da una alternativa de solución al problema planteado, " Supervisión de parámetros de proceso basado en comunicación Bluetooth y equipo móvil".

La supervisión de los parámetros de procesos da una visión y alerta del comportamiento que están teniendo los equipos principales ante posibles anomalías observadas en las lecturas de los parámetros. Así se puede evitar paradas de emergencia no programas de la planta o lo contrario que conlleven al menor tiempo. En el caso de pruebas de comisionamiento se pueden evitar posibles daños irreversibles de los equipos principales que provoquen un retraso considerable en el avance del proyecto. Tanto las paradas de la planta como el daño a los equipos principales son costos que se suman a la pérdida de producción o que se esperaba producir ante un retraso en el avance del proyecto.

El operador en la sala de control tiene la oportunidad de ver el comportamiento del equipo principal a través de la lectura de parámetros que se leen de los equipos de instrumentación instalados en campo. Estos equipos de instrumentación se montan en las líneas de proceso antes o después del equipo principal, o en el mismo equipo. Pero la limitación es que el operador solamente ve al equipo principal como una representación gráfica en el HMI. Se puede colocar una cámara de video en el campo apuntando al equipo incluyendo un micrófono, pero mayormente no sólo es un equipo principal por monitorear. Este planteamiento involucraría un mayor capital de inversión.

Para ello se puede recurrir a los supervisores de campo que están recorriendo todos los puntos de la planta que requieren de mayor atención. Pero como enviarles los datos sin hacer uso de equipos costosos, con poca inversión, poco tiempo de entrenamiento y eliminando la dependencia de tecnologías propias del fabricante de cada

equipo principal o equipo de instrumentación que se utilice. Es evidente el requerimiento de un desarrollo de ingeniería a nivel de software y hardware por lo cual se recurre a una tendencia en la actualidad. "95% de los equipos móviles que compran las personas también son utilizados para el trabajo" [1].

Esta tendencia mencionada es con referencia de una compañía situada en 90 países y con 92,000 empleados. Además el término "Bring Your Own Device" ("Traer tu propio dispositivo" - BYOD) ya se está hablando con mayor frecuencia, y también se ve en el Perú.

Un punto a tratar es que en la etapa del proyecto de una planta industrial se puede aplicar lo que señala el informe. Esto podría ser en la operación, pero el mayor beneficio está en la etapa en comisionamiento. En este último porque es cuando se están realizando las primeras pruebas de los equipos principales y se corre el riesgo de averiarlos generando una pérdida para la compañía, avance en el proyecto y pérdidas por producción programada no entregada.

# CAPÍTULO I
# MARCO TEÓRICO CONCEPTUAL

## 1.1 Comunicaciones Inalámbricas en aplicaciones industriales

Dentro de las tecnologías inalámbricas que aparecen en las plantas industriales como alternativa al cableado clásico o por bus de campo están el Wireless Fidelity / Wireless Local Access Network (WiFi/Wlan), WirelessHart y Bluetooth, existen otras tecnologías más las cuales tienen una presencia mínima en el campo industrial por lo que no se verán. A continuación se desarrollan cada uno para entendimiento de las bondades que ofrecen.

### 1.1.1 Tecnología WiFI/Wlan

Regida bajo las normas de la Institute of Electrical and Electronics Engineers (IEEE) con la numeración 802.11, surgió como alternativa a la comunicación cableada Ethernet 802.03 de la IEEE.

Dentro de las bondades que ofrece están la facilidad de implementación, una mayor libertad en la ubicación de los equipos que conforman la red y como principal característica no requiere el cableado para la transferencia de datos.

Los datos son transmitidos por señales de radiofrecuencia (RF) teniendo un largo alcance sin repetidor hasta los 100 mts aproximadamente. Trabaja en frecuencias cercanas a los 2.4 y 5 Ghz y se encuentran en la banda Industrial, Scientific and Medical (ISM) las cuales no requieren de licencia para su uso.

Para llegar a transmitir las señales existen básicamente tres técnicas y que se especifican en la capa física (capa 1 dentro del modelo Open Systems Interconnection (OSI)):

**a. Espectro de Distribución por Secuencia Directa (Direct-sequence spread spectrum (DSSS))**

Establecido desde el año 1997 trabaja en velocidades de 1 a 2 Mbps para luego en el año 1999 con el estándar 802.11b pasar a soportar 5.5 y 11 Mbps. El DSSS trabaja con canales de 22 Mhz que para el caso de Norte América y en la mayor parte de Europa cubre el ancho de banda desde los 2.4 Ghz hasta los 2.483 Ghz de los cuales se forman tres canales no traslapados. Múltiples copias de la señal utilizando una porción del canal son distribuidas con un patrón de secuencia para que puedan ser decodificadas por el

receptor [2, 3 y 4].

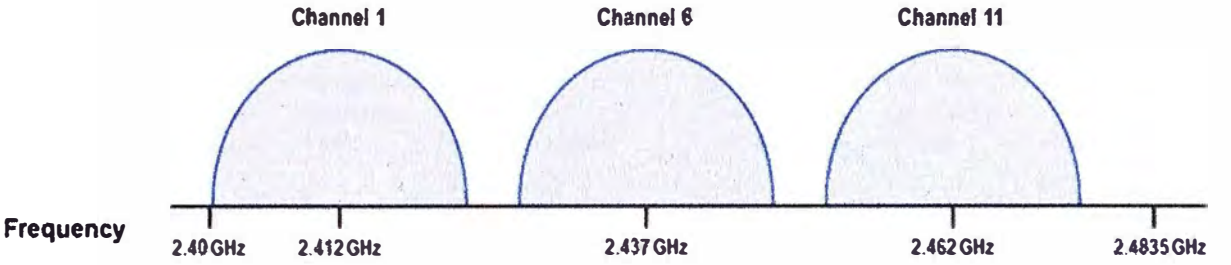



**Fig.1.1** Espectro de distribución por secuencia directa (DSSS) [2].

**b. Espectro de Distribución por Saltos de Frecuencia (Frecuency Hopping Spread Spectrum (FHSS))**

Soporta velocidades de 1 Mbps hasta 2 Mbps y está presente solamente en el protocolo 802.11 original. Esta técnica se basa en saltos o cambios de la frecuencia a usar bajo un patrón establecido. Para el caso de Norte América y Europa el rango utilizado es de 2.402 a 2.480 Ghz de los cuales se obtienen 79 canales no traslapados. Cada uno de los canales con un ancho de 1 Mhz. En la Fig.1.2 se observan los saltos repetitivos que realiza la señal en el espectro de frecuencia.

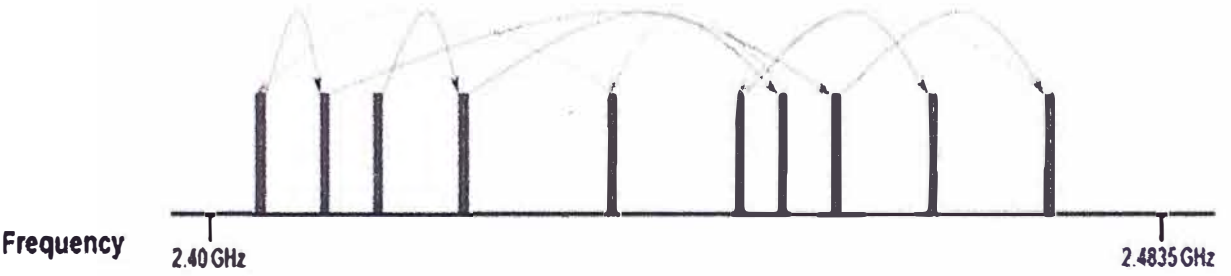



**Fig.1.2** Espectro de distribución por saltos de frecuencia (FHSS) [2].

La secuencia de los saltos de frecuencia que realiza es repetida por el receptor para la decodificación de los datos [1, 2, 3].

**c. Multiplexación por División de Frecuencia Ortogonal (Orthogonal Frecuency División Multiplexing (OFDM))**

Basado en el uso de múltiples portadoras de tipo ortogonales entre sí mismas. Esta característica principal permite que la transmisión de la señal sea de mayor velocidad que las técnicas predecesoras y que sea resistente a atenuación por interferencia. Dentro de los protocolos que adoptan esta técnica está el 802.11a hasta los 54 Mbps (en la banda ISM de 5 Ghz), 802.11g hasta los 54 Mbps y 802.11n hasta los 300 Mbps presente actualmente en el mercado (en la banda ISM de 2 o 5 Ghz) [2, 3 y 4].

En las Tablas N° 1.1 y 1.2 muestran las técnicas que se aplican para determinado

protocolo.

**TABLA N° 1.1** Estándares de las capas físicas de la IEEE 802.11 [4].

| Standard | Date issued | Available bandwidth (MHz) | Unlicensed frequency of operation (MHz) | No. of nonoverlapping channels * | Data rate per channel (Mbps) | Compatibility |
|---|---|---|---|---|---|---|
| 802.11 | 1997 | 83.5 | 2.4 to 2.4835 DSSS, FHSS | 3 indoor or outdoor | 1, 2 | 802.11 |
| 802.11a | 1999 | 300 | 5.15 to 5.35 OFDM (orthogonal frequency division multiplexing) 5.725 to 5.825 OFDM | 4 indoor 4 indoor or outdoor 4 outdoor | 6, 9, 12, 18, 24, 36, 48, and 54 | Wi-Fi5 |
| 802.11b | 1999 | 83.5 | 2.4 to 2.4835 DSSS | 3 indoor or outdoor | 1, 2, 5.5, and 11 | Wi-Fi |
| 802.11g | 2003 | 83.5 | 2. 4 to 2.4835 DSSS, OFDM | 3 indoor or outdoor | 1, 2, 5.5, 6, 9, 11, 12, 18, 24, 36, 48, and 54 | Wi-Fi at 11 Mbps and below |

*Nonoverlapping channels have frequency bands that do not overlap. Their operation can take place in the same area simultaneously.

**TABLA N° 1.2** Estándares de las capas físicas de la IEEE 802.11 [4].

| Standard | Date issued | Scope |
|---|---|---|
| 802.11c | 2003 | Bridge operation at 802.11 MAC layer |
| 802.11d | 2001 | Physical layer: Extend operation of 802.11 WLANs to new regulatory domains |
| 802.11e | Ongoing | MAC: Enhance to improve quality of service (QoS) and enhance security mechanisms |
| 802.11f | 2003 | Recommended practices for multivendor access point interoperability |
| 802.11h | 2003 | Physical or MAC: Enhance IEEE 802.11a to add indoor and outdoor channel selection and to improve spectrum and transmit power management |
| 802.11i | Ongoing | MAC: Enhance security and authentication mechanisms |
| 802.11j | Ongoing | Physical: Enhance IEEE 802.11a to conform to Japanese requirements |
| 802.11k | Ongoing | Radio resource measurement enhancements to provide interface to higher layers for radio and network measurements |
| 802.11m | Ongoing | Maintenance of IEEE 802.11-1999 standard with technical and editorial corrections |
| 802.11n | Ongoing | Physical or MAC: Enhancements to enable higher throughput |

### 1.1.2 Tecnología WirelessHart

Desarrollado como una alternativa a las tecnologías de comunicación inalámbrica en el mercado industrial y con las características de la comunicación Highway Addressable Remote Transducer Protocol (HART), este protocolo trabaja en los 2.4GHz de la banda ISM es decir no requiere licencia de uso.

**a. Historia**

Las normas para esta tecnología fueron iniciados en el año 2004 por un grupo dentro del formado HART Communication Foundation (HCF). Las compañías que destacan en este grupo son Emerson, ABB, Endress+Hausser, Pepperl+Fuchs y Siemens quienes formaron la organización Wireless Industrial Technology Konsortium

(WiTECK) para desarrollar una tecnología inalámbrica con aplicación en el área industrial. Posteriormente en el año 2007 se incluye dentro de los estándares del HART v.7.0., y en el año 2010 es aprobado por el International Electrotechnical Commission (IEC) convirtiéndose en el primer estándar internacional de comunicación inalámbrica con el código IEC 62591 [5 y 6].

**b. Características**

Dentro de ellas, las más resaltantes son:

- Opera en la banda ISM de 2.4Ghz bajo el estándar IEEE 802.15.4.
- Incluye la técnica de saltos de canal, disminuyendo las interferencias en el medio y generación de una lista de canales no recomendados (Backlisting).
- Soporta una red de tipo malla la cual se organiza y recupera por sus propios medios a través de un administrador central de la red. Ante una falla en unos de los equipos, el administrador central se encarga de seleccionar otra ruta alternativa en la transferencia de datos.
- Las redes permiten una escalabilidad para la adición de nuevos equipos.
- Dentro de las topologías a implementar están la de tipo estrella, racimo o malla.
- Basado en la técnica DSSS con una velocidad de hasta los 256 kbps.
- La red utiliza la norma IEEE 802.15.4
- Utiliza la tecnología Time División Múltiple Access (TDMA) que permite una mejor comunicación gestionando los periodos de ausencia de señales.
- Cada equipo presente en la red también actua como ruteador encaminando los datos que recibe. Ello permite un aumento en el nivel de redundancia de la red.
- Soporta las actividades de monitoreo y control de los equipos. Incluye: monitoreo del proceso, monitoreo del ambiente (como por ejemplo temperatura interna), administración de la energía, cumplimiento de normas establecidas, administración de activos, mantenimiento predictivo, diagnósticos avanzados y un control de lazo cerrado en terreno.

  Permite la migración de equipos que trabajan actualmente bajo protocolo HART a WirelessHart [6 y 7].

### 1.1.3 Tecnología Bluetooth

**a. Concepto**

Esta tecnología es de tipo inalámbrica y de libre uso. En la actualidad hay miles de empresas que sus equipos tienen esta tecnología. Ofrece un bajo consumo de energía y sus señales de RF opera alrededor de los 2.4GHz que se encuentra disponible para un uso libre (Banda ISM) [8].

**b. Historia**

Inició en año 1994 como una alternativa para la comunicación cableada de los accesorios utilizados con los teléfonos móviles. Los ingenieros de la empresa Ericsson apuntaban a una comunicación inalámbrica de bajo consumo de energía y alta vulnerabilidad ante interferencias. Es así que surge el Bluetooth como una solución, pero para que sea adoptada por otras empresas también debía de ser un estándar abierto, es decir que no amerite una licencia para su uso y aplicación. Es así que en el año 1998 se crea el Bluetooth Special Interest Group (SIG) conformado por Ericsson, Intel, IBM, Nokia y Toshiba. En 1999 se publica la versión 1.0 de las especificaciones del Bluetooth, para el 2007 se tenía la versión 2.0+EDR con miles de miembros del grupo [8]. En la Fig.1.3 se observan los tiempos que transcurridos entre los lanzamientos de las versiones de la plataforma.

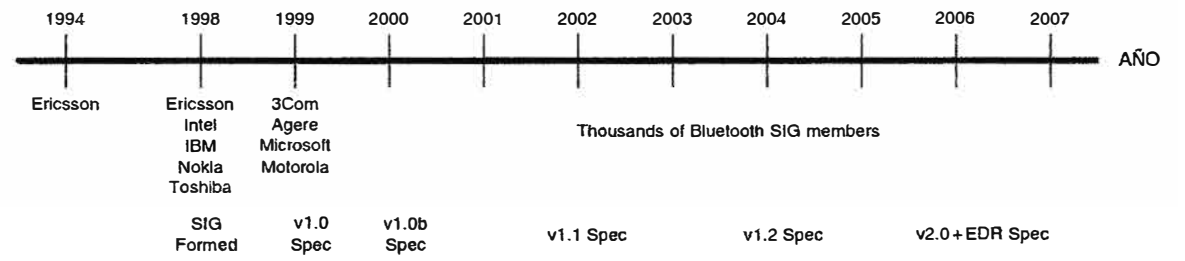



**Fig.1.3** Cronología del Bluetooth SIG [8].

En la actualidad ya se habla de la versión 4+LE lanzada el año 2010, que dentro de sus principales características adicionales tiene una mayor velocidad en la transferencia y protocolo para obtener un bajo consumo de energía (LE), esta última permitiendo su uso en módulos electrónicos alimentados con baterías (pilas) tipo botón (ejemplo CR2032), anulando así el problema de la energía [9].

**c. Características**

- Las especificaciones de las versiones del Bluetooth son abiertas, es decir no presentan derechos de autor por lo que su aplicación puede ser implementada en forma libre.
- Opera con frecuencia de radio de corto alcance para asegurar el bajo consumo de energía dependiendo. Dependiendo de la clase puede llegar hasta una distancia de 100m aproximadamente. En la Tabla N° 1.3 se presentan las diferentes clases respecto a la potencia de transmisión de los equipos con la tecnología Bluetooth.

**TABLA N° 1.3** Clases por potencia de transmisión en la tecnología Bluetooth [10].

| Power class | Transmission power level | Advertised range |
|---|---|---|
| 1 | 100 mW | 100 meters |
| 2 | 2.5 mW | 10 meters |
| 3 | < 1 mW | < 1 meter |

- Soporta la transmisión tanto de voz como datos, el primero más aplicado en los accesorios de manos libres para teléfonos móviles.
- Todas las versiones de la tecnología son compatibles con sus predecesoras.
- Funciona en la banda ISM.
- Respecto al punto anterior existen otras tecnologías que también utilizan esta banda. Para ello la tecnología Bluetooth cuenta con un diseño robusto para minimizar los efectos de las interferencias de otras señales.
- Está ampliamente difundido a nivel mundial y presente en una gran mayoría de equipos no sólo en los accesorios para teléfonos móviles que fue su primera aplicación. Un ejemplo en las casas u oficinas: computadores, impresoras, teléfonos móviles, escáneres, etc., y en el campo industrial: Programmable Logic Controller (PLC), Distributed Control System (DCS), transmisores, etc.

En la Fig. 1.4 se muestra las aplicaciones de la tecnología en los diferentes campos.

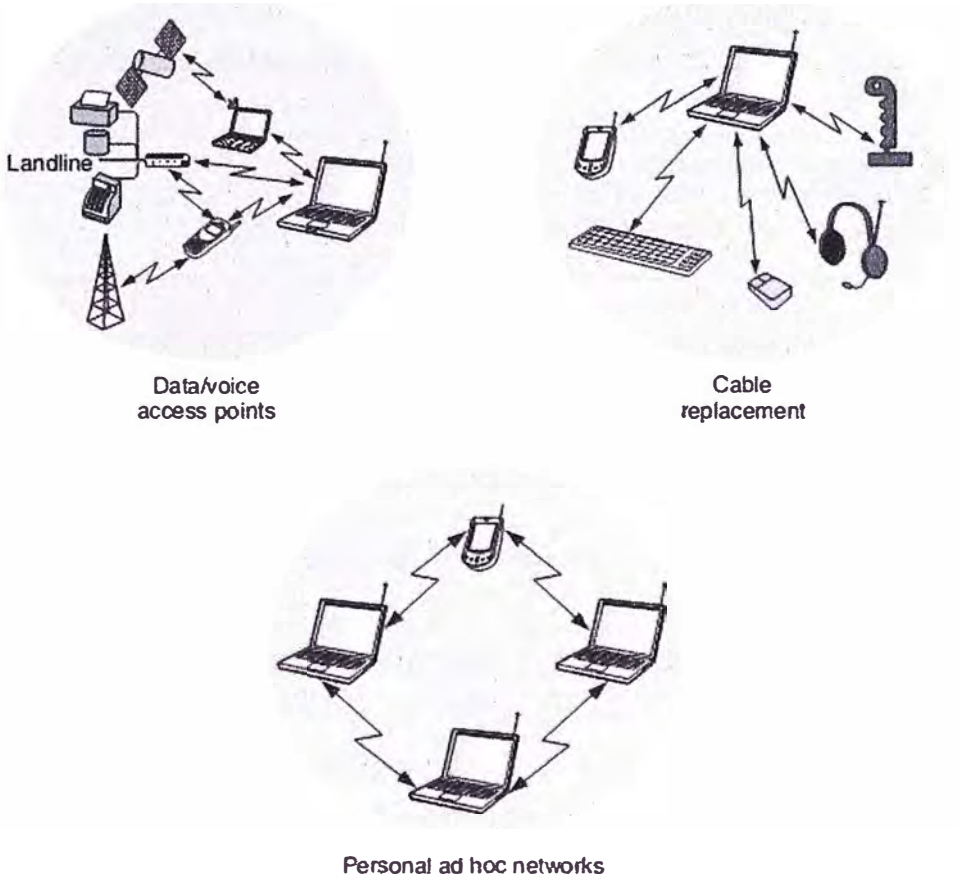



**Fig.1.4** Aplicaciones de la tecnología Bluetooth [8].

**d. Especificación del Bluetooth**

Establece como punto principal el aseguramiento de la compatibilidad entre las diferentes empresas que conforman el Bluetooth SIG, desde las características de las señales de RF hasta las aplicaciones o programas, es decir desde la capa física hasta la capa de aplicación bajo el modelo OSI. En la capa de aplicación también se encuentra la Pila de Protocolos del Bluetooth.

Dentro de esta especificación se encuentran:

**d.1 Especificación del Núcleo (Bluetooth Core Specification)**

Establece la arquitectura del Bluetooth, las características tanto del controlador (Controller) que es el equipo con la tecnología hasta el equipo anfitrión (Host) que aloja el controlador. Incluye las terminologías utilizadas para la comprensión de esta especificación. En la Fig.1.5 se observan las diferentes partes que se requieren para la implementación de la tecnología Bluetooth.

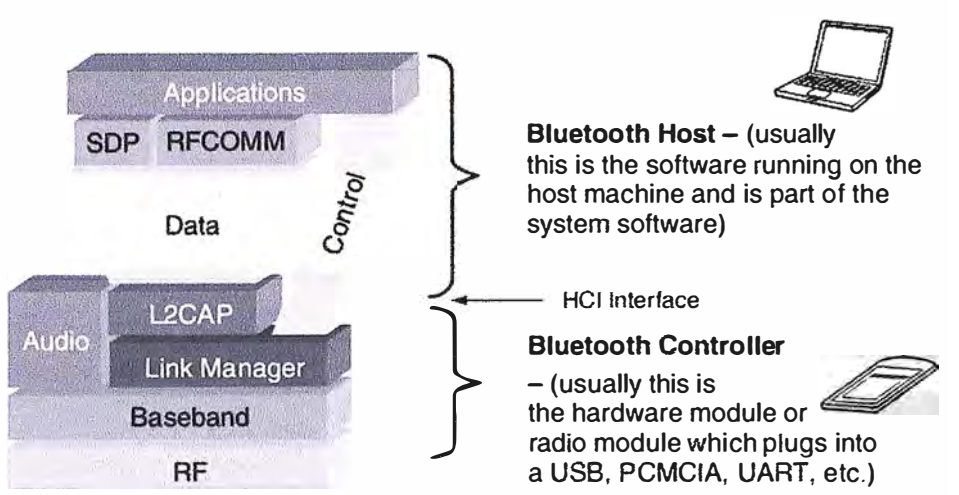



**Fig.1.5** Partes del hardware de la tecnología Bluetooth [8].

**d.2 Especificación del Transporte (Bluetooth Transport Specification)**

Indica los tipos de transporte que se usaran para la transferencia de información entre el equipo anfitrión y el controlador, entre ellos pueden ser por Universal Serial Bus (USB), Personal Computer Memory Card International Association (PCMCIA) o Universal Asynchronous Receiver-Transmitter (UART) [11].

**d.3 Especificación del Protocolo (Bluetooth Transport Specification)**

En ella se indica los tipos de protocolos con que se trabajaran. Entre ellos están el Radio Frequency Communication (RFCOMM) y el OBject EXchange (OBEX). Este punto se amplía posteriormente en Protocolos del Bluetooth.

**d.4 Especificación del Perfil (Bluetooth Transport Specification)**

Indica las posibles aplicaciones que tiene la tecnología Bluetooth, además establece de qué manera se comportara el dispositivo que tenga esta tecnología al momento de comunicarse por este medio. Este punto se amplía posteriormente en Perfiles del Bluetooth [11].

**e. Protocolos del Bluetooth**

En la Fig.1.6. se presenta la pila de protocolos del Bluetooth en donde se indican la mayoría de los protocolos, de los cuales los que están en recuadros sombreados son los que comúnmente se utilizan para la programación con Application Programming Interface (API) bajo plataforma Java con aplicación de la tecnología Bluetooth, Java APIs for Bluetooth Wireless Technology (JABWT) que es materia del presente informe [12].

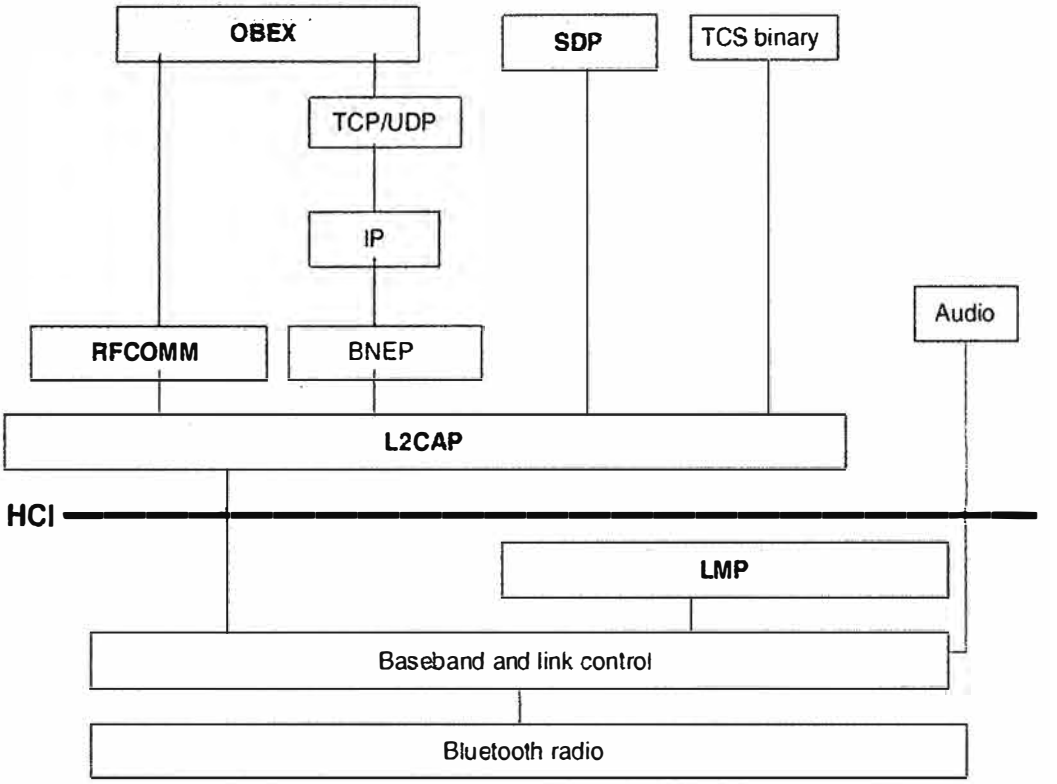



**Fig.1.6** Pila de protocolos de la tecnología Bluetooth [8].

Dentro de la pila de protocolos del Bluetooth se encuentran cuatro que son propios del núcleo de la tecnología:

**e.1 Protocolo RF (Radio Frecuency – Radio Frecuencia):**

La capa respectiva se encuentra en la parte más baja de la pila de protocolos y define la banda de operación ISM en la frecuencia de los 2.4GHz.

**e.2 Protocolo LC (Link Control – Control de Enlace):**

Siendo la capa junto con la banda base (Baseband) los encargados de realizar el conexionado por RF entre los dos dispositivos que se conectan vía la tecnología Bluetooth.

**e.3 Protocolo LM (Link Manager – Administrador de Enlace):**

Especifica el establecimiento del enlace y la configuración respectiva. En este protocolo se ve el aspecto de seguridad como son la autentificación y encriptación.

**e.4 Protocolo L2CAP (Logical Link Control and Adaptation – Adaptación y control del Enlace Lógico):**

La capa respectiva opera como un multiplexador de las conexiones lógicas que realicen capas superiores, se incluye las de audio en el caso de aplicaciones Voice over Internet Protocol (VoIP) como por ejemplo.

En la Fig. 1.7 se aprecian los protocolos mencionados y la interacción con otros

elementos de la pila de protocolos.

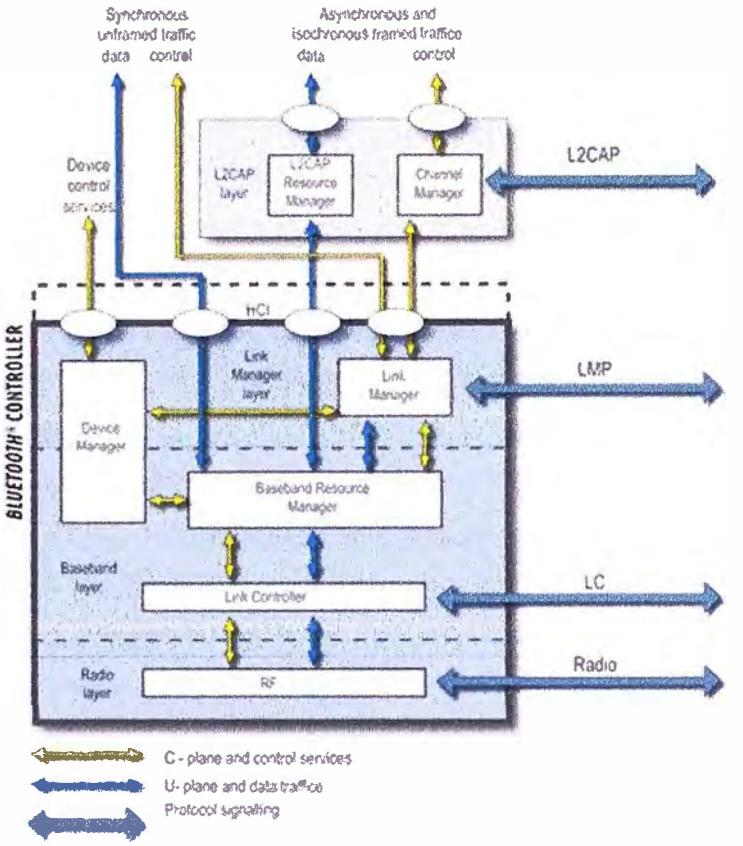



**Fig.1.7** Protocolos de la tecnología Bluetooth [13].

Existe otro protocolo que se utiliza en todas las aplicaciones del Bluetooth y que acompaña a los protocolos del sistema Bluetooth antes mencionados, protocolo SDP.

### e.5 Protocolo SDP (Service Discovery – Descubrimiento de Servicios)

Después de obtener una lista de dispositivos que se encuentran en un rango cercano al equipo anfitrión (Host) el siguiente paso en averiguar qué tipos de servicios cuentan los dispositivos de la lista. Encargado de obtener la lista de servicios que se encuentran en los dispositivos. Con estos datos las aplicaciones pueden saber con qué dispositivos podrán conectarse el equipo anfitrión (Host) a través del controlador. Se entiende como servicio del Bluetooth a toda aplicación que opera como servidor que se encuentra a la espera de una aplicación cliente que solicite este servicio basado en la tecnología Bluetooth [12].

Dentro de los protocolos que existen están los siguientes:

- Protocolo de Control del Transporte de Audio y Video (AVCTP – Audio/Video Control Transport Protocol)
- Protocolo de Distribución del Transporte de Audio y Video (AVDTP – Audio/Video Distribution Transport Protocol)

- Protocolo de Encapsulamiento de Red Bluetooth (BNEP - Bluetooth Network Encapsulation Protocol)
- Protocolo de Intercambio de Objetos (OBEX – OBject EXchange)
- Protocolo de Control para Telefonía (TCP - Telephony Control Protocol )
- Protocolo RFCOMM (RFCOMM - RadioFrecuency COMMunication)

Existen otros protocolos más que incluyendo el OBEX fueron adoptados por el SIG: PPP (Point to Point Protocol), TCP/IP/UDP (TCP/IP Protocols), WAE/WAP (Wireless Application Environment/Wireless Application Protocol). De estos protocolos se explica con más detalle el OBEX y el RFCOMM para interés del presente informe.

### e.6 Protocolo OBEX

Este protocolo tiene sus inicios en la asociación IrDA (Infrared Data Association), la cual autorizó su uso al SIG y fue agregado a la pila de protocolos del Bluetooth. El OBEX se establece teniendo al RFCOMM como base en sus especificaciones, pero además pasó a ser un perfil del Bluetooth [12].

### e.7 Protocolo RFCOMM

Este protocolo emula a la conexión por puerto serial RS-232 entre dos dispositivos pero bajo la tecnología Bluetooth. Ofrece una transferencia de datos al usuario en forma simple y confiable, comparable con el protocolo Transmission Control Protocol (TCP). A diferencia de un puerto serial en donde una sola conexión ocurre, RFCOMM permite que un solo dispositivo servidor pueda recibir varias conexiones de clientes, además un dispositivo como cliente se puede conectar a varios dispositivos tipo servidor.

Existen muchas aplicaciones que usan este protocolo debido a que tiene un soporte ampliamente difundido y disponibilidad de APIs en la mayoría de sistemas operativos. Los equipos que operan con puertos seriales fácilmente pueden adoptar este protocolo. Adicionalmente como punto a resaltar para los propósitos del informe el protocolo RFCOMM es ampliamente utilizado en las aplicaciones bajo plataforma Java [12].

### f. Perfiles del Bluetooth

El perfil (Profile) realiza una selección de protocolos con que opera, lo mismo hace con las características que ofrece, así obtiene un modelo particular para el uso en una aplicación.

Dentro de la gama de perfiles del Bluetooth, éstos se pueden agrupar en cuatro perfiles básicos que estarán conformados por otros perfiles. Estos son: GAP (Generic Access Profile), SPP (Serial Port Profile), SDAP (Service Discovery Application Profile) y GOEP (Generic Object Exchange Profile). En la Fig.1.8 se observan estos perfiles.

Estos cuatro perfiles son también llamados Perfiles de Transporte debido a que otros perfiles llamados Perfiles de Aplicación los toman como base para su elaboración.

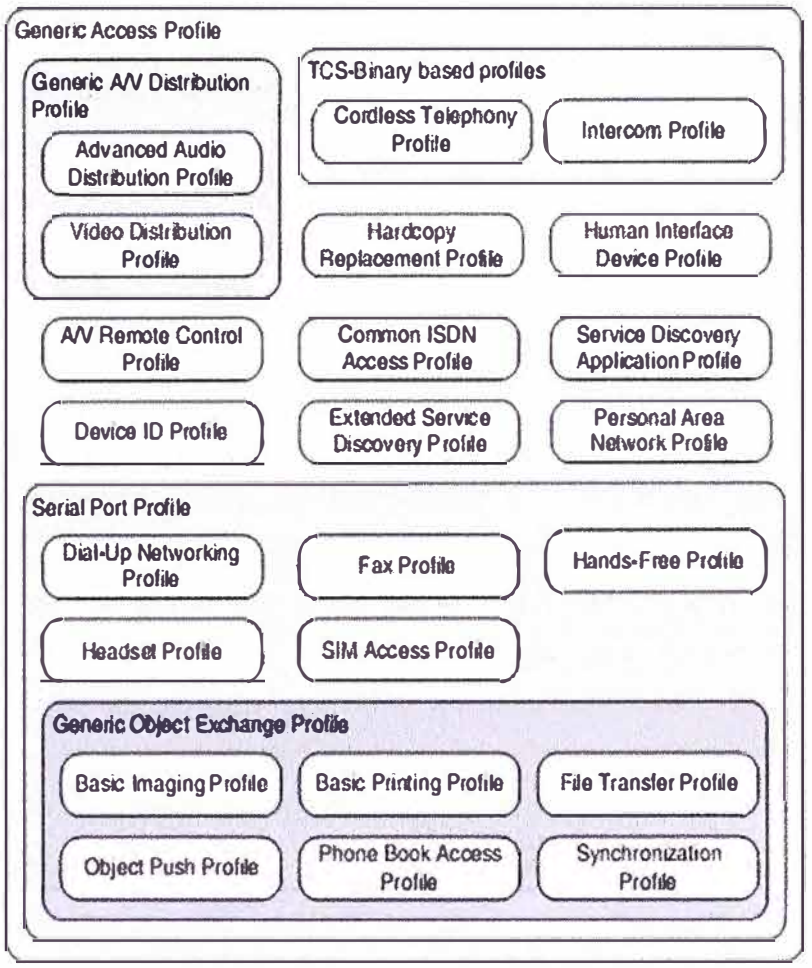



**Fig.1.8** Jerarquía de los perfiles de la tecnología Bluetooth [8].

### f.1 Perfil GAP

Este perfil está presente en todos los demás, es decir a partir de él se desarrollan los otros. Involucra el establecimiento de conexión, descubrimiento de los dispositivos que se encuentran alrededor, administración del enlace, configuración y la seguridad de la misma (autentificación y encriptación).

### f.2 Perfil SDAP

Las aplicaciones se basaran en los protocolos y procedimientos que involucran este perfil apuntando a obtener la lista de servicios que ofrece el dispositivo servidor.

### f.3 Perfil SPP

Establece los requerimientos que tanto el dispositivo servidor como cliente deberán ofrecer para establecer la conexión. Este perfil usa el protocolo RFCOMM para la conexión entre los dos dispositivos involucrados.

### f.4 Perfil GOEP

En un perfil base para la elaboración de otros perfiles de mayor complejidad. Este

perfil usa el protocolo OBEX

## 1.2 Plataformas para teléfonos móviles

Actualmente existe una variedad de opciones de las cuales solamente se detallaran las que tienen mayor presencia en el mercado local. Como punto excepcional se explica la plataforma Java (específicamente la plataforma Java 2 Micro Edition (J2ME)), que no siendo actualmente una plataforma nativa en los teléfonos móviles permite ampliar su uso y aumentar la compatibilidad de las aplicaciones de software a instalar y ejecutar para las demás plataformas.

### 1.2.1 Plataforma y Lenguaje de Programación Java

#### a. Concepto

Es una plataforma con lenguaje de programación orientada a objetos presente en la mayoría de las familias de equipos electrónicos empezando por teléfonos móviles, equipos Personal Digital Assistant (PDA), lavadoras, horno microondas, automóviles, faxes, televisores, computadoras y servidores como algunos de los ejemplos.

Cada uno de los equipos indicados (equipos anfitriones) tiene características particulares además de aplicaciones distintas. Por ello se definieron más plataformas a partir del Java las cuales como las de más presencia en el mercado se tienen: JEE (Java Enterprise Edition), JSE (Java Standard Edition) y JME (Java Micro Edition) [8].

#### b. Historia

Tiene sus inicios en el año 1991 en la compañía Sun Microsystems, Inc con un primer nombre "Oak" ("Roble") y cambiado en el año 1995 como actualmente se conoce "Java".

Apuntando a un lenguaje de programación con plataforma independiente que permita desarrollar softwares a ser utilizados en sus inicios en equipos domésticos como ejemplo lavadoras, hornos microondas y control remoto para televisores. Dentro de los primeros problemas encontrados estaba el proceso de compilación tanto en costo como en tiempo que requería la elaboración de compiladores para cada tipo de CPU (Central Processing Unit) que se encontraban en los controladores y estos a su vez en los equipos electrónicos. Estos problemas fueron solucionados pensando en una plataforma que sea portable (a instalarse en la mayoría de CPU's) y con lenguaje de programación independiente de la plataforma.

Con el desarrollo del World Wide Web (WWW) y la necesidad de aplicaciones portables para la variedad de tipos de computadoras, permitió a Java incrementar su presencia en el mercado.

El crecimiento del mercado con nuevos equipos electrónicos para los usuarios abre un nuevo espacio para las aplicaciones en plataforma Java específicamente para los

teléfonos móviles.

En el año 1999, el alcance que logra la plataforma Java y viendo las capacidades tanto en hardware como en alojamiento de software de los diferentes equipos da pie a la compañía Sun Microsystems, Inc a crear plataformas basadas en Java que permitan cubrir específicamente cada familia de equipos. Estas plataformas permiten una mayor facilidad a los desarrolladores de programas, proveedores de servicios y a los fabricantes de dispositivos. Es así que se presentan las nuevas plataformas con la nueva versión, Java 2 Plataform, Standard Edition (J2SE), Java 2 Plataform, Enterprise Edition (J2EE) y Java 2 Plataform, Micro Edition (J2ME).

Para el año 2006, Sun Microsystems con la finalidad de difundir aún más el uso de las nuevas tecnologías es que establece el Java como un desarrollador de código abierto bajo la GNU General Public License (GNU GPL), siendo GNU un sistema operativo similar al UNIX pero de código abierto. La licencia GNU es la misma que rige para el sistema operativo Linux y se aplica para las tres plataformas mencionadas anteriormente J2EE, J2SE y J2ME [14, 15 y 16]. En la Fig.1.9 se muestra la magnitud de presencia que tiene la plataforma Java.


**Fig.1.9** Java en el mercado mundial [16].

### c. Plataformas del Java

Debido la creciente demanda en el uso de la plataforma Java en las diversas

familias de equipos electrónicos es que Sun Microsystems Inc crea tres plataformas.

- J2SE: Para el desarrollo de aplicaciones destinadas a las computadoras de escritorio.
- J2EE: Para el desarrollo de aplicaciones a ejecutar en servidores y computadoras empresariales.
- J2ME: Para el desarrollo de aplicaciones a instalar en equipos portátiles, entre los cuales se encuentran los teléfonos móviles.

  En la Fig.1.10 se observa las diferentes plataformas con sus respectivos VM.

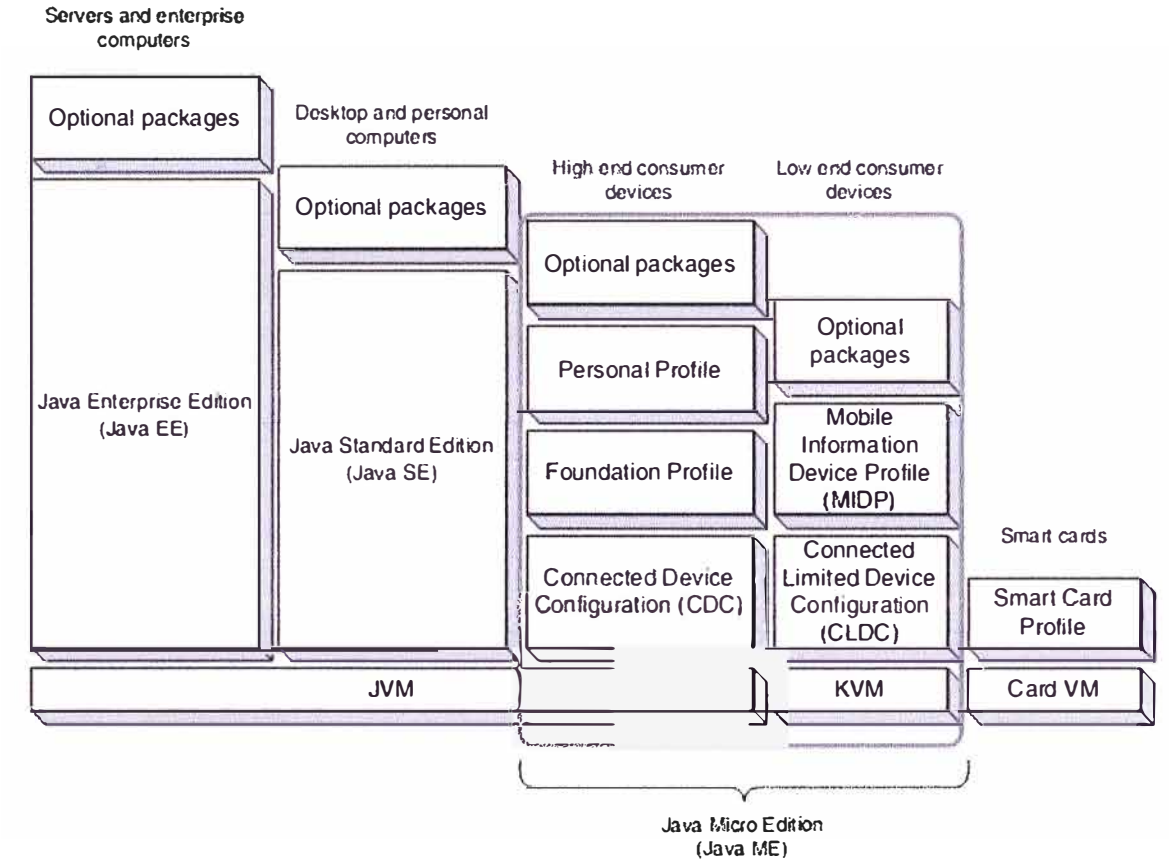



**Fig.1.10** Plataformas Java [8].

De estas 3 plataformas y como parte importante del presente informe se desarrolla la plataforma J2ME.

**d. Plataforma J2ME**

Dirigida para su aplicación en equipos portátiles o de requerimientos mínimos de procesamiento y hardware dentro de ellos la capacidad gráfica. Dentro de sus características heredadas de la plataforma J2SE se tiene: portabilidad del código, programación orientada a objetos y facilidad en el desarrollo de aplicaciones.

Para lidiar con las limitaciones de capacidad de los equipos anfitriones (Host) es que en esta plataforma se hace uso de las MV's (Máquinas Virtuales) que son softwares

que permiten acondicionar la ejecución de aplicaciones para que operan en una plataforma distinta a la existente.

En la Fig.1.11 se indican los componentes de la plataforma J2ME, entre ellos: las configuraciones, los perfiles y paquetes opcionales.

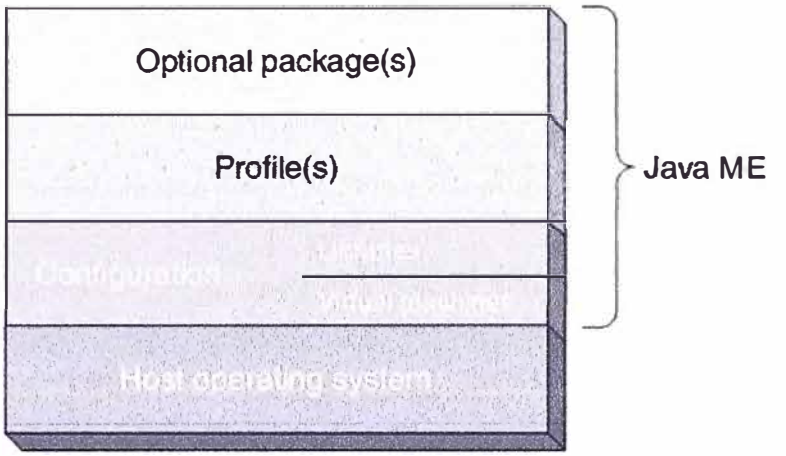



**Fig.1.11** Componentes de la arquitectura Java ME [8].

### d.1 Máquinas Virtuales para J2ME

Para esta plataforma se emplea el Kilo Virtual Machine (KVM) o Compact Virtual Machine (CVM). El KVM que requiere unos pocos kilobytes de memoria a comparación de las MV clásicas que requieren de 1 a 10 megabytes. Para el CVM los requerimientos de memoria van desde 1 megabyte a más. Estos VM se pueden apreciar en la Fig.1.12.

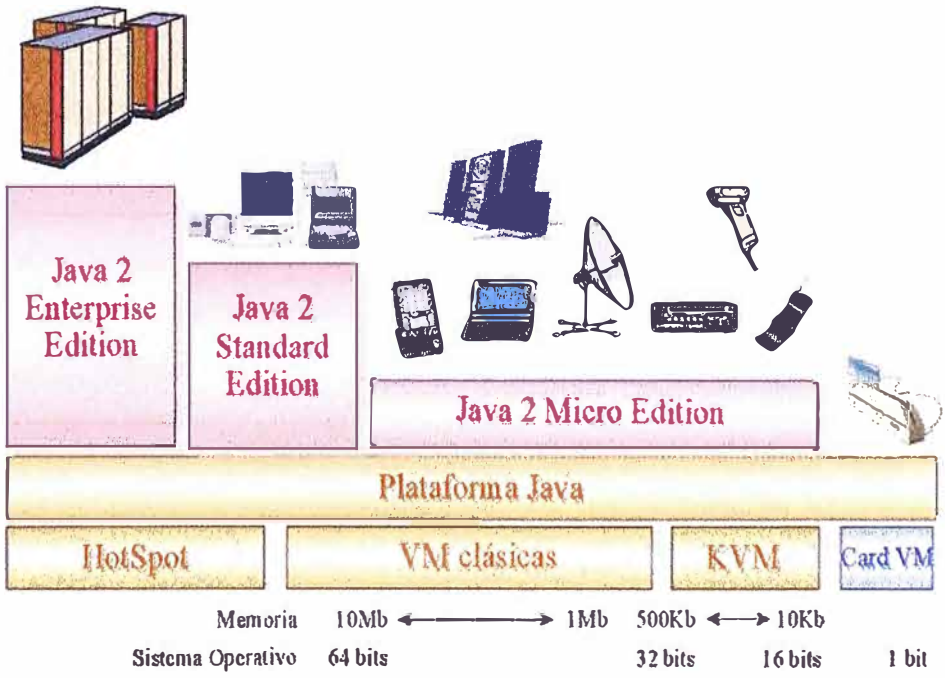



**Fig.1.12** Arquitectura de la plataforma Java 2 [14].

### d.2 Configuraciones

Conformada por una Java Virtual Machine (JVM) y un grupo de librerías en donde se encuentran los APIs. Las configuraciones agrupan la mínima cantidad de APIs básicas que permiten desarrollar aplicaciones a un grupo de equipos con características en común.

Para la plataforma J2ME se tiene 2 tipos de configuraciones: Connected Device Configuration (CDC) y Connected Limited Device Configuration (CLDC) que se detalla a continuación.

#### d.2.1 CDC - Connected Device Configuration (Configuración para Dispositivo con Conexión)

Dirigido a equipos de consumo de alta funcionalidad las cuales cuentan con más memoria, procesadores veloces y mayor tiempo de conexión a una red en comparación con la configuración CLDC. Entre los equipos se tiene: decodificadores para televisión (TV) y comunicadores de alta funcionalidad. Trabaja conjuntamente con el CVM [14].

#### d.2.2 CLDC - Connected Limited Device Configuration (Configuración para Dispositivo Limitados con Conexión)

Implementado para equipos de consumo de baja funcionalidad. Entre ellos: organizadores personales y teléfonos móviles los cuales presentan limitaciones en la memoria, velocidad del procesador, duración de la batería y de comunicación a una red por lapsos. Incluye la KVM por requerir el mínimo de memoria comparado con el CVM [14].

### d.3 Perfiles

Es el complemento de la configuración respectiva. Agrupa los APIs necesarios para completar una plataforma de mayor funcionalidad con respecto a la configuración para un grupo más específico de dispositivos.

En la plataforma J2ME entre los principales perfiles se tienen cinco: Foundation Profile (FP), Personal Profile (PP), Remote Method Invocation Profile (RMIP), Personal Digital Assistant Profile (PDAP) y Mobile Information Device Profile (MIDP). De estos cinco los dos últimos funcionan con la configuración CLDC, mientras que los tres primeros lo hace con la configuración CDC [14].

En la Fig.1.13 se muestran la arquitectura que tiene la plataforma J2ME.

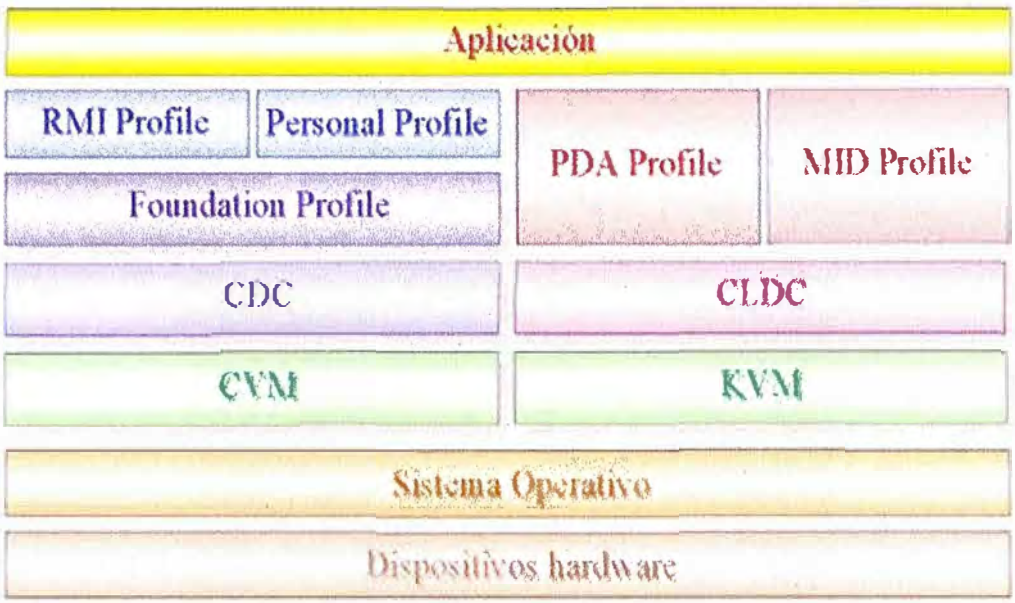



**Fig.1.13** Arquitectura de la plataforma J2ME [14].

**d.3.1 FP (Foundation Profile - Perfil Fundación)**

Para equipos embebidos que no presentan interface para usuario pero si una capacidad de red.

**d.3.2 PP (Personal Profile - Perfil Personal)**

Para equipos como por ejemplo agendas personales y comunicadores de alta funcionalidad, y consolas de juego con interface de usuario y conexión a Internet.

**d.3.3 RMIP (Remote Method Invocation Profile - Perfil para Invocación en forma remota)**

Requiere una implementación bajo el perfil FP.

**d.3.4 PDAP (Personal Digital Assistant Profile - Perfil para Asistente Digital Personal)**

Este perfil es similar al PP pero para equipos de la misma familia con baja funcionalidad.

**d.3.5 MIDP (Mobile Information Device Profile - Perfil para Asistente Digital Personal)**

Fue el primer perfil creado. Entre los equipos figuran teléfonos móviles y organizadores personales de funcionalidad básica. Junto con la configuración CDLC ofrecen una alta funcionalidad que comprende una interface de usuario, capacidad de red y almacenamiento continuo de datos. Así el MIDP ofrece un ambiente para la ejecución de aplicaciones en equipos móviles para información.

**d.4 Paquetes Opcionales**

Estos paquetes conforman el complemento al conjunto perfil-configuración y

apuntan a cubrir las necesidades de programación para la aparición de nuevas tecnologías entre ellas: la tecnología Bluetooth inalámbrica, multimedia, mensajería y conectividad.

### 1.2.2 Plataforma Android


**Fig.1.14** Plataforma Android en el mercado mundial [17].

#### a. Concepto

Es una plataforma desarrollada para equipos móviles bajo el kernel Linux. Tiene licencia de uso libre y es de código abierto. En la Fig. 1.14 se observa la dimensión de aceptación en el mercado de los equipos móviles sobretodo. Dentro de los primeros equipos que adoptaron esta plataforma fueron los teléfonos móviles, pero en la actualidad pasaron a las tablets, reproductores mp3, cámaras digitales, laptops y computadoras [18,19 y 20].

#### b. Historia

En el año 2003, se crea la compañía Android Inc, teniendo como trabajo el desarrollo de aplicaciones para equipos móviles. En el año 2005 Google adquiere la compañía. En el año 2007 se anuncia la creación de la Open Handset Alliance (OHA) una organización que tenía entre sus miembros compañías de gran presencia en el mercado entre ellos a Google. Seguidamente OHA hace el anuncio del lanzamiento del Android v1.0 y en el año 2008 sale al mercado el primer teléfono móvil con plataforma Android, el "Dream G1" de la empresa HTC [21, 22 y 23].

Actualmente se tiene Android v.4.2 "Jelly Bean" y suman 400 millones de equipos con sistema operativo Android. En la Fig.1.15 se muestra la evolución de las versiones.

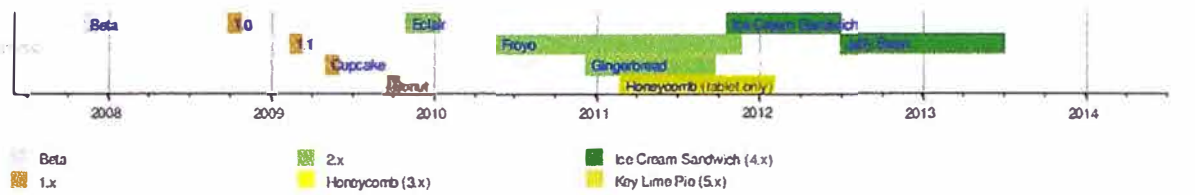



**Fig.1.15** Cronología de las versiones de la plataforma Android [24].

**c. Arquitectura**

**c.1 Sistema Operativo:**

Basado en el kernel del sistema operativo Linux ofreciendo una interface de bajo nivel con el hardware, la administración de la memoria y el control de proceso.

**c.2 Middleware:**

Un entorno que permite la ejecución de aplicaciones Android. Agrupa tanto al VM Dalvik y librerías de núcleo.

**c.3 Aplicaciones móviles claves:**

Entre ellas: email, Short Message Service (SMS), Personal Information Manager (PIM) y Navegador web.

**c.4 Librerías API para desarrollo de aplicaciones móviles:**

Conformado por librerías de código abierto: SQLite, WebKit y OpenGL ES.

En la Fig.1.16 se observa las diferentes partes de la arquitectura de la plataforma Android.

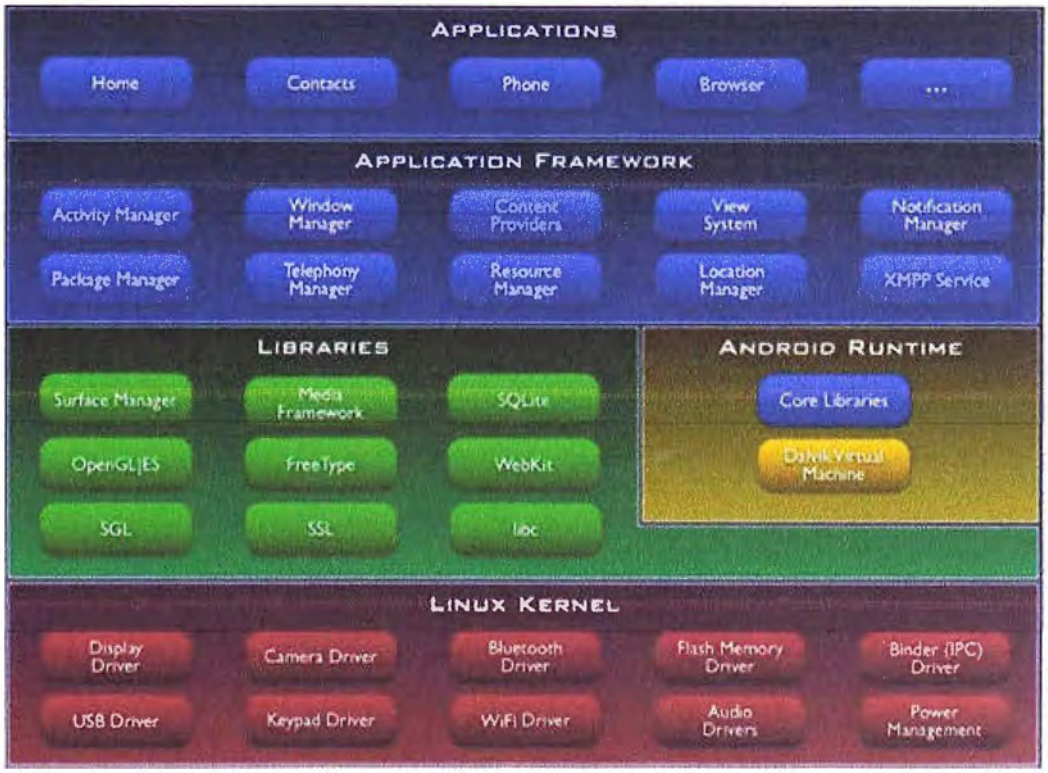



**Fig.1.16** Arquitectura de la plataforma Android [18].

**d. Características Adicionales**

- Soporta formatos de audio, video e imágenes (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF).
- Soporta telefonía GSM, Bluetooth, EDGE, 3G y WiFi dependiendo del equipo móvil.
- Gestionamiento de la cámara, GPS y acelerómetro dependiendo del equipo móvil.
- Soporta pantalla táctil.
- Un único pago de $25 para el registro de desarrolladores y la distribución de las aplicaciones a través del Google Play (tienda virtual de aplicaciones Android).

**e. OHA (Open Handset Alliance)**

Conformada por empresas prestadoras de servicios, fabricante de teléfonos móviles, fabricantes de semiconductores, compañías desarrolladoras de software y compañías comercializadoras.

OHA es el desarrollador del Android, la primera plataforma para equipos móviles completamente gratuita y de código abierto.

El objetivo de la organización es la promoción y desarrollo de la plataforma Android. Así también establecer estándares abiertos para las aplicaciones, hardware de los equipos móviles y todo lo que involucre el desarrollo de nuevas tecnologías [25]. La Fig.1.17 muestra parte de las compañías que conforman la OHA.


**Fig.1.17** Principales compañías de la alianza OHA [18].

### 1.2.3 Plataforma iOS

**a. Concepto**

Es una plataforma diseñado exclusivamente para equipos de la compañía Apple:

iPhone (teléfono móvil), iPod touch (reproductor de música), iPad (Tablet) y en la actualidad el Apple TV (televisor) con las últimas versiones.

El iOS ejecuta aplicaciones basado en el lenguaje Objective C, muy similar al lenguaje C [27].

**b. Historia**

En año 1980 un grupo de desarrolladores inician los primeros pasos del lenguaje Objective C como un proyecto de agregar más bondades al lenguaje C. En año 1996 la compañía Apple compra los derechos de NeXT Computer propietaria del sistema operativo NeXTSTEP generado a partir del lenguaje Objective C. Es así que el NeXTSTEP es renombrado a MacOS X en el año 2000. En el año 2006, se crea el iOS a partir del MacOS X. Seguidamente en el año 2007 se lanza al mercado el iPhone con iOS y que en la actualidad tiene el iOS v.6.0 [28]. La Tabla N° 1.4 indica los cambios de versión que fueron ocurriendo.

**TABLA N° 1.4** Cronología de las versiones de la plataforma iOS [29].

Discontinued | Current | Beta

| Version | Build | Release date | Highest version for |
|---|---|---|---|
| 3.1.3 | 7E18 | February 2, 2010; 2 years ago | iPhone (original); iPod Touch (1st generation) |
| 4.2.1 | 8C148 | November 22, 2010; 2 years ago | iPhone 3G; iPod Touch (2nd generation) |
| 5.1.1 | 9B206 | May 7, 2012; 6 months ago | iPod Touch (3rd generation); iPad (original) |
| 5.1 | 10A406E | September 24, 2012; 2 months ago | Apple TV (2nd & 3rd generation) |
| 6.0.1 | 10A523<br>10A525 | November 1, 2012; 25 days ago | iPhone 3GS, iPhone 4, iPhone 4S, iPhone 5; iPod Touch (4th & 5th generation); iPad 2, iPad (3rd generation), iPad (4th generation), iPad Mini |
| 6.1 (Beta 2) | 10B5105c | November 12, 2012; 14 days ago | iPhone 3GS, iPhone 4, iPhone 4S, iPhone 5; iPod Touch (4th & 5th generation); iPad 2, iPad (3rd generation), iPad (4th generation), iPad Mini |

**c. Arquitectura**

Conformado por cuatro capas:

**c.1 Core OS**

Basado en el kernel OS X y Mach 3.0, tiene como características resaltantes: uso de sockets BSD (Berkeley Software Distribution), sistema de archivos y permite interactuar con equipos externos incluyendo unidades de almacenamiento [30].

**c.2 Core Services**

Dentro de los servicios que ofrece están: uso de la estructura Core Location, soporta integración a redes, multitareas (multi-threading), cuenta con directorio y soporta fuentes SQLite y XML [30].

### c.3 Media

Permite soportar imágenes, audio y video basados en CoreAudio, OpenGL OpenGL ES, Core Audio, Quartz y Core Animation. Adicionalmente soporta PDF [30].

### c.4 Cocoa Touch

Funciona con los marcos UIKit y GameKit que incluye trabajar: vía multitarea, con memoria temporal de datos (Clipboard), ventanas, eventos de contacto y movimiento, sensores de proximidad, cámara, micrófono, librería de fotos, detección de movimientos y navegador entre otras bondades [30].

En la Fig.1.18 se aprecia las partes que conforman la arquitectura de la plataforma iOS.

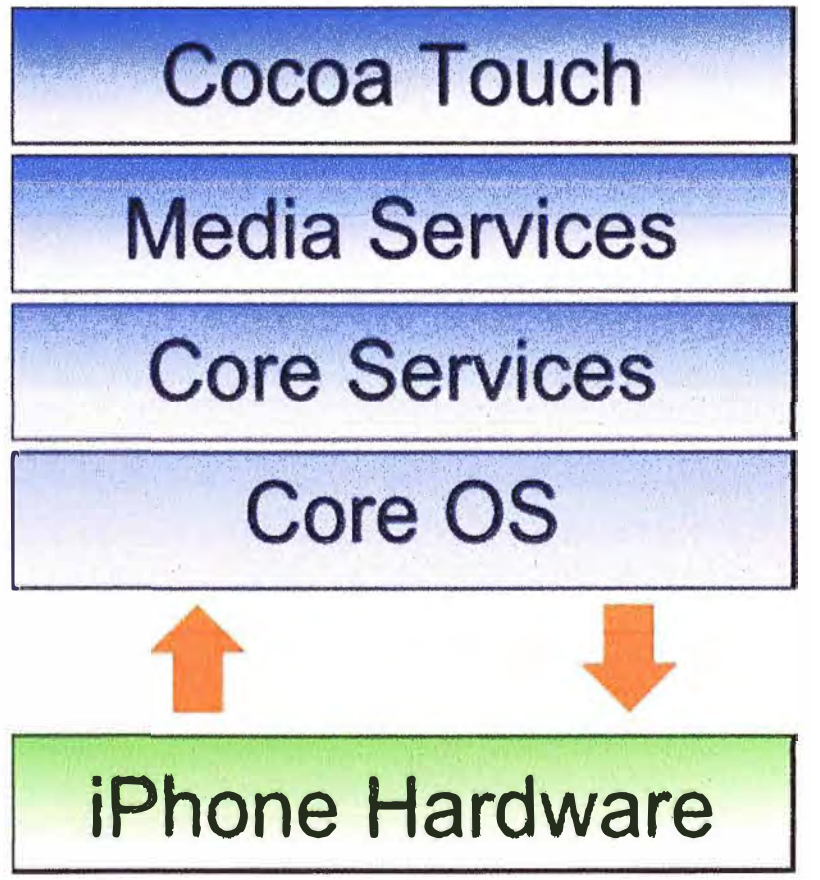



**Fig.1.18** Arquitectura de la plataforma iOS [31].

### d. Características Adicionales

- La plataforma iOS no es de código abierto.
- Actualmente Apple está otorgando licencias a determinadas compañías que en algunos casos es por demandas [32].
- Para los desarrolladores, es requerimiento realizar un pago anual de $99 que permite distribuir la aplicación desarrollada en la tienda de Apple (App Store).

### 1.2.4 Plataforma Blackberry

**a. Concepto**

Es una plataforma de código cerrado teniendo como propietario a la compañía Research In Motion (RIM). Desarrollado para equipos móviles desde sus inicios hasta la actualidad agregándose en estos últimos años los equipos tablets. En cuanto a los teléfonos móviles Blackberry, son lo que tienen mayor presencia en el campo empresarial.

**b. Historia**

En el año 1999, la plataforma hace su aparición con el localizador RIM 850 en la versión 1.0 y programado con lenguaje C++. En el año 2000, RIM anuncia sus trabajos en obtener un software de desarrollo para aplicaciones Java y para el año 2001 se lanzan los primeros localizadores RIM con kernel basado en J2ME. Para el año 2002, se presentan al mercado los primeros teléfonos móviles Blackberry 5810 en la versión 3.6. Debido al desarrollo del que se alcanzó en el hardware y a los servicios adicionales que venían con un teléfono como eran el BIS (Blackberry Internet Solution y BES (Blackberry Enterprise Solution), la compañía resuelve llamar a la plataforma Blackberry. En la actualidad para el próximo año se espera el lanzamiento de la versión 10.0 basado en el OS QNX Neutrino real-time y con kernel QNX Neutrino RTOS Secure [33, 34, 35, 36, 37 y 38].

**c. Arquitectura**

La plataforma Blackberry está soportada por un JVM desarrollado en lenguaje C++. Las partes que lo conforman la arquitectura se observan en la Fig.1.19.

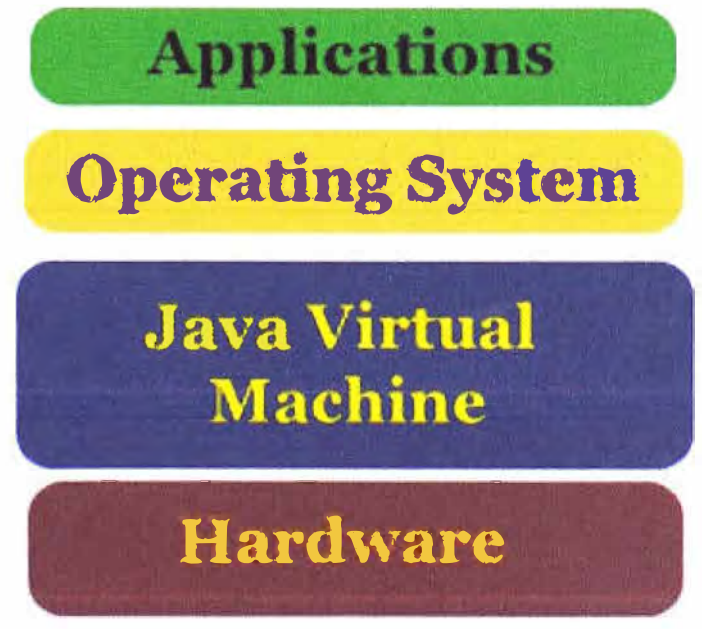



**Fig.1.19** Arquitectura del Blackberry OS [39].

Como se observa en la Fig.1.20, tanto la configuracion, los perfiles y API's adicionales del Blackberry OS se basan en la plataforma Java (CDLC, MIDP, PDAP, JSR y MIDlets), especificamente J2ME, en donde Java Specification Requests (JSR) indica los requerimientos para nuevas tecnologias a ser manejadas por Java.

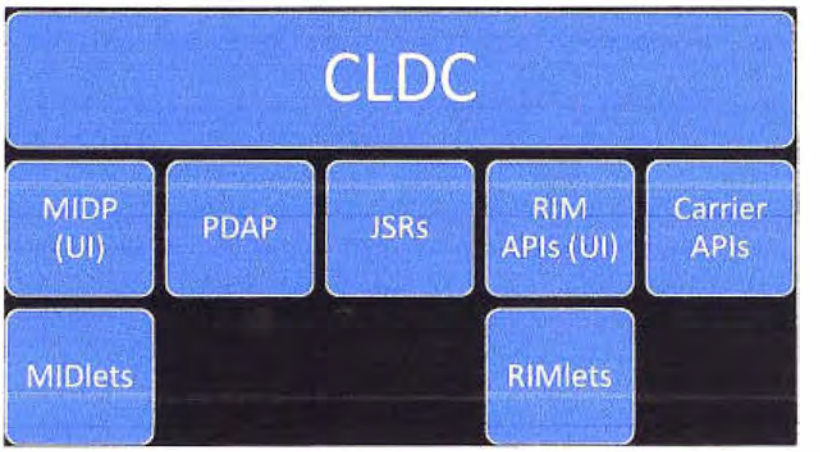



**Fig.1.20** Arquitectura de APIs para Blackberry [35].

Para el Blackberry 10 según lo indicado es de la siguiente forma:

- Plataforma Blackberry basado en OS QNX Neutrino RTOS.
- Kernel QNX Neutrino RTOS.

La Fig.1.21 muestra la nueva arquitectura que vendrá con la versión Blackberry 10.

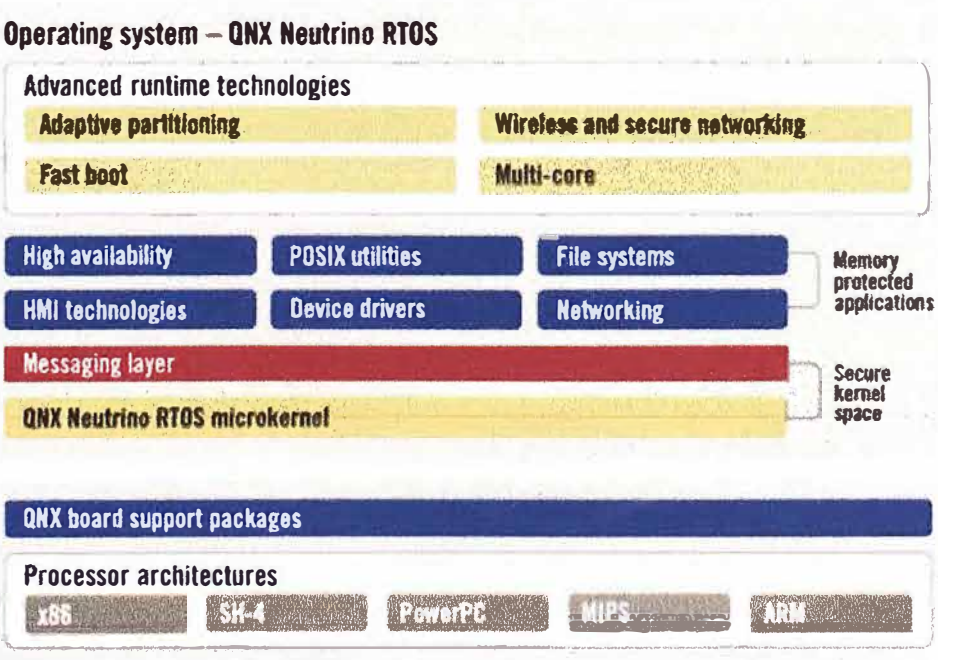



**Fig.1.21** Arquitectura del OS QNX Neutrino RTOS [40].

### d. Características Adicionales

De las mencionadas además se tiene:

- Operating System (OS) multitarea y multiproceso.
- Los teléfonos móviles vienen con JVM.
- Los APIs del Blackberry permiten la interacción con el hardware como por ejemplo: Luz de fondo, acelerómetro, pantalla táctil, GPS, Wii y audio.
- Cuenta con servicios de conectividad adicionales propios de Blackberry como el BIS y BES.
- El pago para registrarse en la tienda de Blackberry (Blackberry App World) es de $0.00.

## 1.3 Etapas de un Proyecto para una Planta Industrial

Los factores que tienen mayor importancia en un proyecto son: costo, tiempo, calidad, alcance y lo más importante, la seguridad. Posteriormente y como finalidad del presente informe se darán mayor resalte a la mejora del costo y el tiempo.

Dentro de las etapas de un proyecto en forma general se mencionaran las principales.

### 1.3.1 Estudio de Factibilidad

Para el desarrollo de un proyecto se requiere la inversión de capital que junto con un informe sustentatorio del monto requerido forman parte del inicio de un proyecto. Esta etapa es importante para el inicio del proyecto y tiene por finalidad obtener la aprobación de la inversión a través de la recolección de información sustentatorio. El aseguramiento del buen término de esta etapa hace que esta se desarrolle en forma lenta. En esta etapa los administradores del proyecto analizaran los recursos humanos, materiales, limitaciones de fondos y cambios de políticas en el tiempo de vida del proyecto entre algunos puntos [41].

Como principales puntos a cumplir se tiene los estudios para:

- Determinación del fondo requerido para la inversión.
- Capital de inversión inicial
- Gastos de pre-operatividad
- Entidades inversoras en donde se incluye sus estados financieros.

Esta etapa deberá de abarcar los siguientes puntos:

- Factibilidad técnica
- Establecimiento de una organización
- Capacidad de gerenciamiento
- Aspectos comerciales
- Factibilidad financiera

### 1.3.2 Ingeniería

En esta etapa a partir de la información que se tiene del estudio se inicia la elaboración de documentos, diagramas y planos del diseño de la planta.

En esta etapa cualquier cambio que se realice genera que la etapa de construcción requiera mayor inversión financiera e incremento en el tiempo del proyecto.

Dentro de los objetivos principales está:

- Completar el informe del proyecto que contiene toda la información que se tiene a la fecha.
- Elaboración de costos y tiempos estimados del proyecto.
- Generación de la documentación que son base para el inicio de las etapas de construcción y procura en donde se incluye las licitaciones de servicios.

Como actividades a ejecutar se tienen:

- Elaboración de un resumen final del proyecto.
- Ejecución de estudios técnicos entre ellos: de topología, suministro de materiales y de mercado.
- Elaboración de documentación principalmente diagramas, especificaciones técnicas, planos, metrados, costos estimados de construcción y operación CAPEX (CAPital EXpenditure) y OPEX (OPerating EXpenditure), planes de construcción, planes de operación y cronogramas.
- Obtención de la aprobación por parte del usuario del informe final del proyecto [42].

### 1.3.3 Procura y Licitación

Con la documentación elaborada en la etapa de Ingeniería inicia la compra de los elementos que forman parte de la planta, así también las licitaciones de servicios.

Dentro de los puntos principales de esta etapa se tienen:

- Compra de equipos y materiales requeridos según la ingeniería.
- Selección de las compañías que ofrecen sus servicios en la etapa de construcción.

Como actividades principales están:

- Solicitud de cotización para la compra de equipos y materiales.
- Licitación los servicios a requerirse en la etapa de construcción.
- Evaluación de las propuestas tanto para la compra de equipos y materiales como también de servicios. Esta evaluación tiene dos partes: la técnica y comercial. Esta actividad concluye con asignación de las compras y contrataciones de las compañías [42].

### 1.3.4 Construcción

En esta etapa se da inicio a la ejecución de los trabajos de construcción y de la instalación de las partes que conformaran la planta.

Dentro de esta etapa se elabora y actualiza continuamente la siguiente documentación para un mejor control del avance:

- Cronograma de construcción.
- Esquema de los recursos humanos
- Cronograma de la llegada de los equipos y materiales.

Las actividades que se desarrollarán son:

- Trabajos de largo y corto plazo.
- Suministro de equipos y materiales.
- Coordinaciones permanentes con las compañías prestadoras de servicios, incluye la supervisión de la calidad del servicio.
- Supervisión de la seguridad la cual incluye charlas periódicas de inducción al personal de campo.
- Supervisión permanente del avance en campo y seguimiento según la documentación de ingeniería [42].

### *1.3.5 Precomisionamiento*

Involucra las siguientes actividades las cuales se realizan antes del montaje final del equipo:

- Inspección de los equipos para una verificación de las especificaciones requeridas y aseguramiento de su funcionamiento.
- Verificación y adecuación de los parámetros configurables que requieren los equipos.
- Elaboración de documentación acreditando las actividades anteriores [42].

### 1.3.6 Comisionamiento

En esta etapa se evaluan los equipos y materiales montados en campo, esto incluye el aseguramiento en la ejecución de las actividades de mantenimiento y operación después de la puesta en marcha de la planta.

La finalidad de esta etapa es el aseguramiento en el funcionamiento de los equipos y estado de los materiales para la puesta en marcha de la planta.

Dentro de las actividades se encuentra:

- Elaboración de un registro de las instalaciones concluidas.
- Inspección de posibles fallas o desviaciones según la documentación de ingeniería y propias del fabricante. Incluye el seguimiento a la solución de las observaciones encontradas.
- Generación o agrupación de la documentación a ser utilizada en las actividades de mantenimiento y operación.
- Realización de pruebas del funcionamiento de los equipos que en algunos casos

requiera la presencia del material o fluido [42].

Esta etapa da paso a la puesta en marcha de la planta para luego entrar a la etapa de operación que incluye la ejecución periódica de mantenimiento de los equipos.

## 1.4 Programación en J2ME

Se basa en la generación de Midlets que es una aplicación Java basada en la configuración CLDC y perfil MIDP. Es decir es una aplicación orientada al uso en equipos móviles de bajas funcionalidades y capacidad.

### 1.4.1 Midlet

Las aplicaciones Midlet están conformadas por dos archivos de extensión JAR (Aplicación en sí con sus recursos requeridos) y JAD (Descriptor de la aplicación). La elaboración del Midlet se puede realizar vía línea de comando o a través de un entorno visual.

#### a. Estructura del Midlet

Dentro de las cinco etapas del ciclo de vida del Midlet que se muestra en la Fig.1.22, se ampliará más los estados que presenta el Midlet en la etapa de Ejecución.

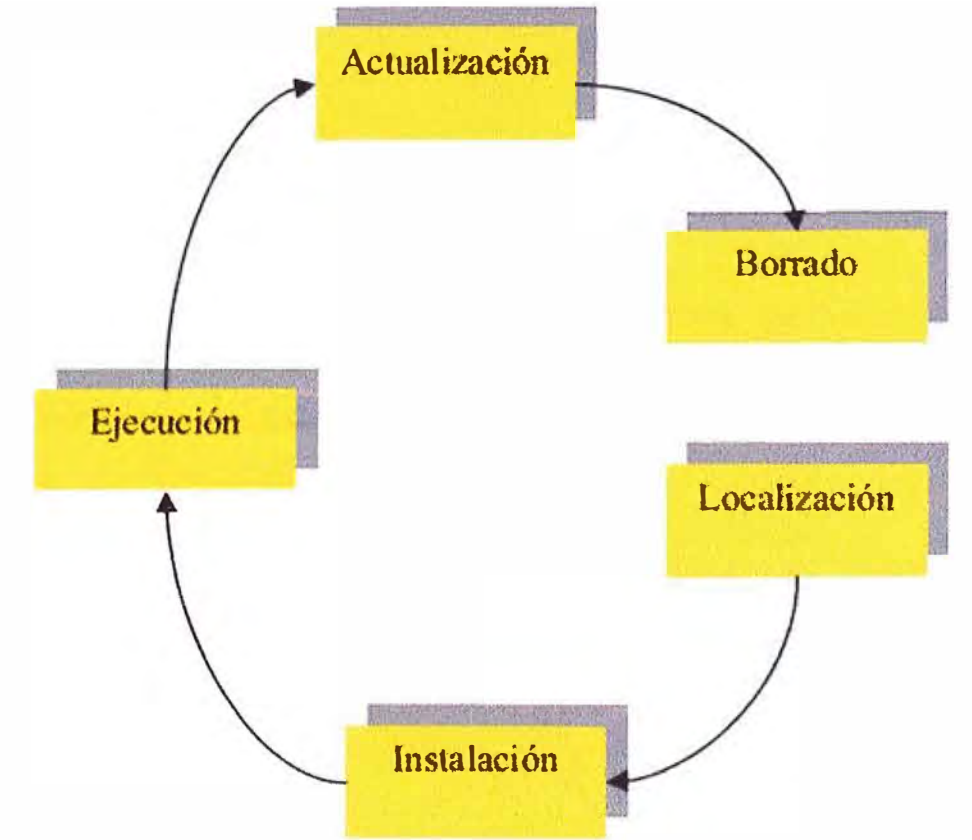



**Fig.1.22** Ciclo de vida del Midlet [14].

Los estados que presenta la etapa de Ejecución son: Activo, Pausa y Destruido. Estos tres estados como se muestran en la Fig.1.23 están relacionados con tres métodos

que se usan en la elaboración del Midlet que son: startApp(), pauseApp() y destroyApp().

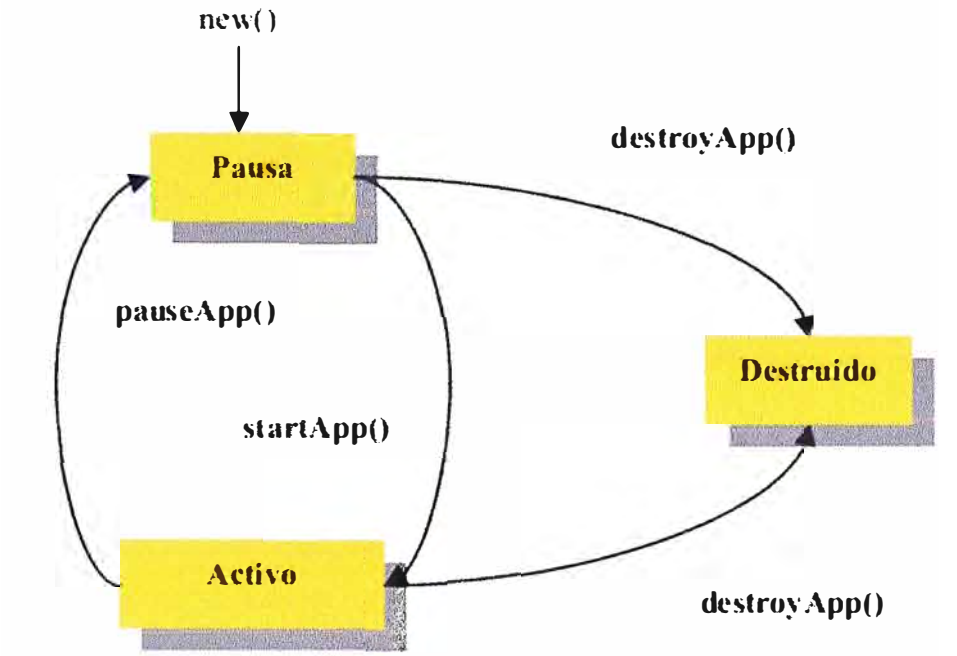



**Fig.1.23** Estados del Midlet en la etapa de Ejecución [14].

En la primera parte de la elaboración básica del Midlet se tiene la invocación a los paquetes que se usan. Estos paquetes contienen las clases y métodos que se usan en el Midlet.

Código en referencia:

```
Import javax.microedition.midlet.*;
```

Seguido va la declaración del nombre que lleve el Midlet, que tipo de clase es y se indica que es una subclase de Midlet con la palabra "extends". Cabe indicar que después de esta línea también se puede declarar las variables y objetos que se usen en el Midlet o bien en el método constructor.

Código en referencia:

```
public class MidletTitu extends Midlet {
```

En la parte del método de constructor se inician los objetos que irán con el Midlet ya sea en su ejecución o incluidos en las pantallas que se muestran en el equipo móvil.

Código en referencia:

```
public MidletTitu () {
  /* Constructor del Midlet
  */
}
```

Los siguientes métodos son referenciados a los estados del Midlet.

Código en referencia:

```
public startApp () {
/* Código a ejecutarse cuando el Midlet se encuentra en estado de Activo.
*/
}

public pauseApp () {
/* Código a ejecutarse cuando el Midlet se encuentra en estado de Pausa.
*/
}
public destroyApp () {
/* Código a ejecutarse antes que el Midlet se cierre o pase al estado de Destruido.
*/
}
```

### 1.4.2 Interfaces gráficas del usuario

Basado en el paquete "javax.microedition.ldcui". Este paquete contiene las clases a utilizar para mostrar diferentes elementos en la pantalla. La Fig.1.24 muestra las diferentes clases que conforman la jerarquía de la clase Display.

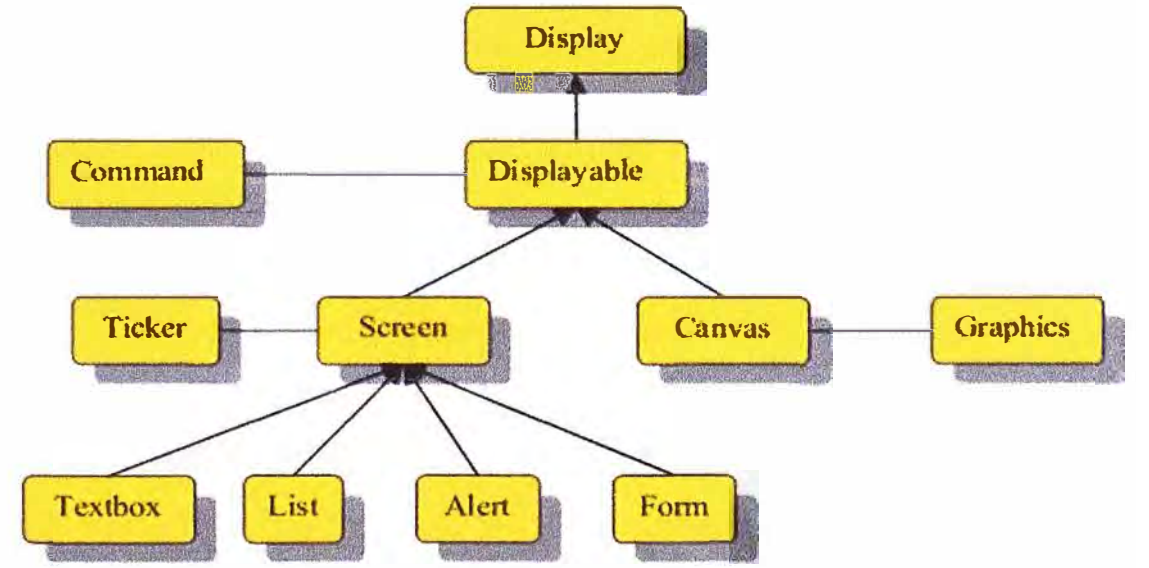



**Fig.1.24** Jerarquía de clases de Display. [14].

#### a. Clase Display

Se encarga de administrar la pantalla del Midlet. Para el uso de la pantalla se tiene que generar un objeto Display de la siguiente forma:

Código en referencia:

```
Display screen = Display.getDisplay(this);
/* "screen" es el nombre del objeto, "getDisplay()" es el método que permite hacer
una referencia a la pantalla del Midlet, y "this" representa al Midlet.
*/
```

**b. Clase Displayable**

Es la representación de las pantallas a usar en el equipo móvil.

**c. Clase Command e Interface CommandListener**

Usado para detectar y ejecutar una acción determinada a partir de una selección hecha en la pantalla. Para que la selección de esta clase se detecte se usa la interface CommandListener. También se requiere implementar el método commandAction() en donde está el código a ejecutar después de detectada la selección.

### 1.4.3 Interface de usuario de alto nivel

Las clases siguientes son subclases de la clase Screen, y esta a su vez de Displayable.

**a. Clase Alert**

Utilizada para avisar al usuario de un evento ocurrido. Esta clase ocupa toda la pantalla y dependiendo de los argumentos utilizados puede solamente mostrar un texto o también una figura.

**b. Clase List**

Para mostrar diferentes opciones en la misma pantalla se utiliza esta clase. Presente tres tipos: IMPLICIT, EXCLUSIVE y MULTIPLE. De estas tres la que permite solamente la selección de una opción es EXCLUSIVE.

**c. Clase Form**

Esta clase permite que se agreguen más clases del tipo Item. La Fig.1.25 muestra las subclases de Item.

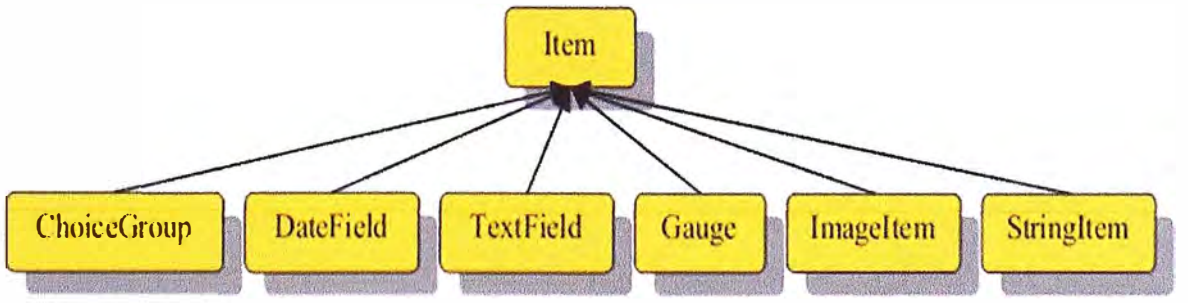



**Fig.1.25** Jerarquía de la clase Item [14].

**d. Clase ChoiceGroup**

Es subclase de Item y que a diferencia de la clase List, utiliza la misma pantalla de la clase Form para mostrarse.

### 1.4.4 Interface de usuario de bajo nivel

Para poder generar este tipo de interface se usa la clase Canvas que permite manipular los elementos mostrados en la pantalla hasta el nivel de pixeles. Dentro de los

métodos a utilizar los siguientes son los más resaltantes.

**a. Método Paint()**

El código de este método permite dibujar figuras en la pantalla cada vez que sea invocada la clase a la que pertenece este método.

**b. Método Repaint()**

En la estructura de una subclase de tipo Canvas, existen otros métodos que detectan el uso de las teclas de equipo móvil. A partir de ello se puede utilizar el método repaint(). Este método permite ejecutar nuevamente el método paint() para dibujar, modificar o borrar toda o parte del gráfico mostrado en la pantalla.

### 1.4.5 Clase Thread

Este tipo de clase requiere de un objeto con interface Runnable. Esta clase permite ejecutar el código en paralelo con la ejecución del Midlet. Se pueden utilizar tantos objetos de la clase Thread como sean necesarios para dar rapidez a la ejecución del código del Midlet.

**a. Método Run()**

Este método se debe implementar cada vez que se tenga una clase con interface Runnable.

### 1.4.6 Interfaces

Las interfaces que es muy diferente a las interface de usuario que se comentaron, son un medio para poder agregar funciones a las clases permitiendo utilizar otros métodos para el manejo del Midlet. Las interfaces indicadas anteriormente son CommandListener y Runnable. Para la interface CommandListener, permite el uso del método commandAction() para poder detectar y ejecutar eventos relacionados con la selección en la pantalla. Para la interface Runnable, permite el uso del método run() para ejecutar un código en paralelo con otros códigos que están ejecutándose.

### 1.4.7 Clase TimerTask

Esta clase se encuentra incluida en el paquete "java.util" y junto con el objeto de clase timer ejecutan una tarea en forma periódica.

# CAPÍTULO II
# PLANTEAMIENTO DE INGENIERÍA DEL PROBLEMA

## 2.1 Descripción del problema

En forma general en todos los proyectos de plantas industriales ante un evento negativo (desperfecto técnico o error humano) podría significar un aumento considerable de los factores costo y tiempo. Esto se puede dar en las etapas de ingeniería, construcción o comisionamiento.

Por un lado en la etapa de construcción ante un cambio de ingeniería (planos o documentos) puede terminar en una doble compra de un equipo principal (bomba, chancadora, o por ejemplo un molino) por no cumplir con las nuevas especificaciones técnicas. Esto puede suceder teniendo al tiempo como el factor que más es afectado. Se tendría que cotizar el equipo nuevamente, realizar los trámites de compra y esperar a que llegue el equipo que en el menor de los casos es como mínimo cuatro meses si no se encuentra disponible en los almacenes del proveedor.

En el caso de la etapa de ingeniería los tiempos perdidos son menores ya que la solución es la generación de nueva documentación de ingeniería como por ejemplo ante una decisión equivocada. Esta etapa es de las primeras y se podría manejar los tiempos por estar más alejado de la entrega del proyecto.

En el caso de la etapa de construcción el montaje de los equipos principales es una de las primeras actividades después de finalizado los trabajos civiles y estructurales. De ocurrir un evento negativo, como caso extremo se tendría que iniciar una nueva compra. En este punto los factores costo y tiempo son afectados por igual pero manejables.

Para la etapa de comisionamiento los casos ya son críticos, esto sumado a que los equipos son principales y se encuentran en la ruta crítica del cronograma de proyecto. Estos eventos negativos definitivamente afectarían el costo y tiempo del proyecto. En el caso del costo sería similar al que se tiene en la etapa de construcción. Pero para el factor tiempo es uno de los eventos más críticos que se tienen en todo el ciclo del proyecto. Como es sabida la etapa de comisionamiento se realiza previo a la puesta en marcha de la planta y la entrega de la misma, es decir el cierre del proyecto. Al ser un equipo principal todo el tiempo de entrega de un nuevo equipo recaería en la ruta crítica y

así en la entrega de la planta. En este punto también se tendría que sumar el costo de producción programada que se está perdiendo.

Por lo indicado anteriormente, el aseguramiento de que un evento negativo no ocurra en la etapa de comisionamiento permitirá una entrega de la planta dentro del costo y tiempo programado, o en caso contrario con leves atrasos debido a eventos inevitables.

## 2.2 Objetivos del Informe

Entre los principales se tienen:

- Aseguramiento de la entrega de la planta en el tiempo y costo estimado según lo planteado en el cronograma y estimaciones de costos desarrollados en la etapa de ingeniería.
- Apertura de nuevas tendencias de tecnologías para su aplicación en los proyectos de plantas industriales.

La Fig.2.1 muestra lo que se ve actualmente en las plantas industriales y como sería con el logro de los objetivos del presente informe.

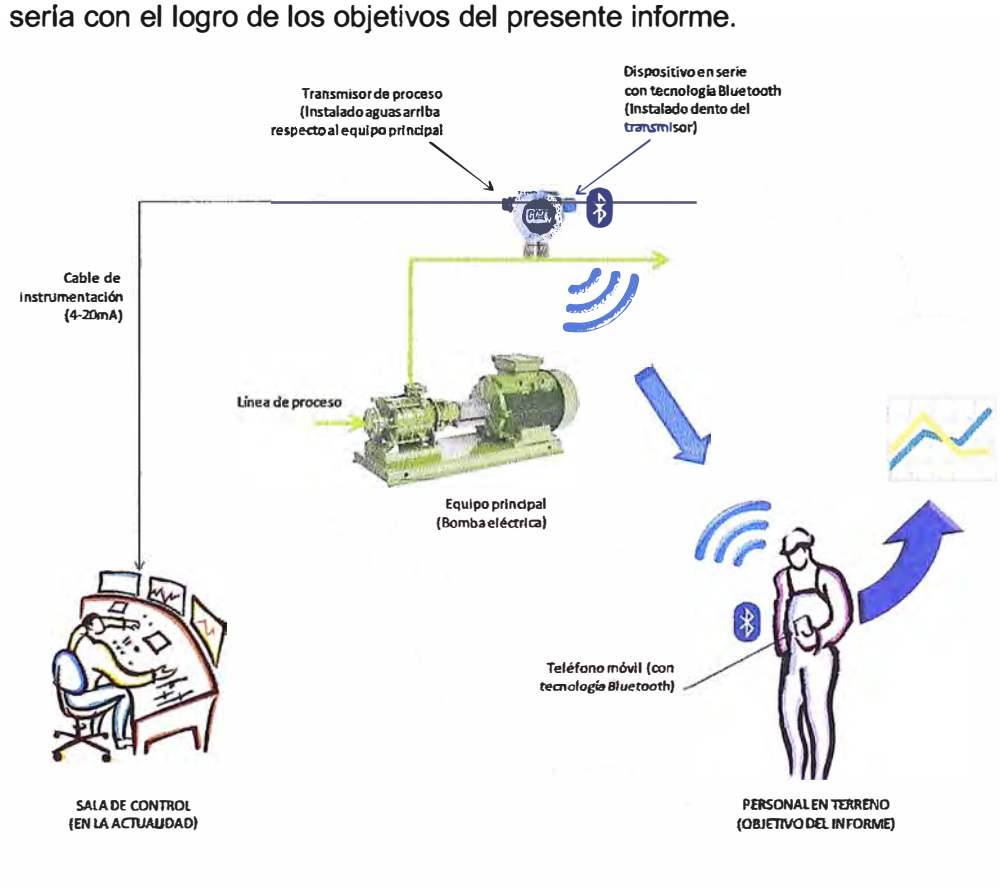



**Fig.2.1** Esquema del monitoreo de parámetros [Elaboración propia].

## 2.3 Evaluación del problema

El aseguramiento de la entrega de la planta parte de minimizar el impacto negativo en los principales factores de un proyecto que son el costo y tiempo. Dentro de un

cronograma o estimación de costos los equipos principales tienen el mayor impacto sobre estos factores. En el caso del cronograma la mayoría de los equipos principales se encuentran en la ruta crítica, principalmente por tener el mayor tiempo de fabricación que se refleja en el mayor tiempo de entrega. Para la estimación de costos los equipos principales son los de mayor monto ya sea por la tecnología que tienen, cantidad o costo del material con que se fabrica, o complejidad en su funcionamiento, esto último ligado al costo total de horas-hombre de ingeniería por parte del fabricante.

Como se mencionó anteriormente, la etapa de comisionamiento es una de las más críticas en el tiempo de vida del proyecto, y el cuidado con los equipos principales tiene la mayor prioridad. En esta etapa a manera de prueba se ponen en funcionamiento los equipos por primera vez, toda acción de prevención garantizará aún más el término de las pruebas en forma exitosa.

Los equipos principales normalmente vienen con instrumentos de protección, como en el caso de las bombas en donde los motores vienen con sensores de temperatura instalados interiormente en las bobinas y/o chumaceras, o en el caso de los molinos donde los instrumentos de protección no solo se encuentran en el motor sino también en el sistema de lubricación para los engranajes como son los sensores de temperatura o de presión. Todos estos instrumentos de protección normalmente se encuentran formando lazos en el circuito eléctrico del accionamiento de los equipos principales que ante cualquier lectura o detección fuera de los parámetros normales de operación se activan originando una parada de emergencia. Esto permite realizar una revisión y encontrar el motivo que generó la parada y posteriormente darle solución.

Todos estos instrumentos de protección que normalmente vienen con los equipos principales así como los diagramas de conexionado de los circuitos eléctricos en la mayoría de los casos son considerados por el mismo fabricante para controlar que posibles acciones humanas o estado de los equipos lleguen a impactar negativamente en las pruebas y en el proyecto en su conjunto.

En la etapa de ingeniería no solo se cuenta con los instrumentos de protección de los equipos principales. También en esta etapa con la finalidad de proteger a los equipos principales se adicionan otros más ya sea incorporándolos o en las líneas de proceso antes y después del mismo. Estos equipos en la mayoría de casos son transmisores entre los que se podría mencionar para el caso de las bombas los transmisores de flujo y presión, y en el caso del molino pueden ser los transmisores de vibración.

Las señales de estos transmisores son de 4-20mA y llegan hasta la sala de control donde se muestran las lecturas en el HMI.

La lectura de estos parámetros de presión, temperatura o vibración es importante,

refleja el comportamiento interno que está teniendo el equipo principal. En el caso de las bombas, los daños en la misma por efecto de cavitación ante ausencia intermitente de fluido se refleja en la oscilación de la presión en la línea. Para el molino, el aumento de la vibración en las chumaceras o lecturas fuera del rango de operación en la temperatura del fluido de lubricación pueden afectan su composición, sus propiedades de lubricación y resultando en la fricción de las partes mecánicas de las chumaceras.

El personal encargado de las pruebas en la etapa de comisionamiento deberá de prevenir cualquier percance que pudiera ocurrir y para ello se debe de tomar las medidas que están a su alcance. Estas pruebas se realizan normalmente con los trabajadores de la etapa de construcción, el personal técnico del proveedor y los supervisores del comisionamiento de entre los cuales se encuentra el operador de la sala de control. Para las pruebas, el operador informa al supervisor que se encuentra en campo de las lecturas de los parámetros que envían los transmisores. La intermediación de la lectura por parte del operador podría generar una falla crítica si es que ocurriera un error de lectura o si no se está monitoreando por el HMI un parámetro en particular. Se tiene que recordar que en la etapa de comisionamiento las pruebas se realizan por partes y la puesta en marcha del sistema de control que incluye también los lazos de protección lógica posiblemente no fue habilitado o no pasaron las pruebas de comisionamiento.

## 2.4 Alcance del informe

El presente trabajo se enfoca principalmente en la etapa de comisionamiento debido a que se obtiene el mayor aporte para el desarrollo del proyecto de una planta industrial. Dentro de las actividades a ejecutar se tienen:

- Análisis de los casos críticos que se pueden presentar en los equipos principales.
- Planteamiento de los requerimientos mínimos tanto en hardware como software. Se estudia las tendencias de tecnologías que ofrece el mercado.
- Elaboración de un listado con requerimientos mínimos para una propuesta de solución al problema planteado considerando los costos mínimos y la disponibilidad de tecnología actual tanto por parte de la empresa como del personal de comisionamiento.
- Análisis de las alternativas de solución en la parte del hardware y software.
- Selección de los dispositivos finales requeridos.
- Elaboración de la estimación de costos y cronograma.

## 2.5 Síntesis del informe

El presente informe a partir de los eventos negativos que se pueden presentar en los equipos principales en la etapa más crítica de un proyecto que vienen a ser la de comisionamiento, se plantea una solución considerando el menor costo de inversión

posible tanto en hardware como en software. Asimismo la solución se apoya en las nuevas tendencias de tecnologías disponibles en el mercado que actualmente se están empezando a usar en las plantas industriales y de la disponibilidad de equipos móviles ya sea por parte del personal de comisionamiento o por la empresa.

# CAPÍTULO III
# METODOLOGÍA PARA LA SOLUCIÓN DEL PROBLEMA

## 3.1 Introducción

Específicamente el problema que se está dando es como lograr que el personal de comisionamiento, como por ejemplo el supervisor de campo, quién en el procedimiento de prueba de un equipo principal, tenga la oportunidad de monitorear los datos de proceso de los instrumentos de protección (como transmisores de presión, temperatura o vibración) que están incorporados o alrededor del equipo, y a la vez pueda observar en forma directa el comportamiento físico que tenga el mismo equipo teniendo ambas acciones, la finalidad de tomar medidas de protección en forma inmediata minimizando en un mayor grado posibles daños al equipo principal.

Recurriendo a los procesos de desarrollo de software y debido a la complejidad incierta de la solución al problema es que se aplica el método de diseño "Top-down" o modular. Es un método usado en el desarrollo de software y que consiste en dividir el problema en partes que se pueden realizar de forma más sencilla para unir luego los resultados de cada parte y obtener una solución final [43].

## 3.2 Análisis del problema

Necesariamente el dar la opción al supervisor de monitorear los datos de proceso en campo es a través de los instrumentos de protección que se encuentran incorporados o cercanos al equipo principal. Pero estos instrumentos como son transmisores cuyas señales en la mayoría de los casos son de 4-20mA se transmiten a la sala de control.

El supervisor podría coordinar con el operador para que constantemente informe de la lectura que está obteniendo, pero en la comunicación se pueden presentar errores de lectura por parte del operador o que los datos en algunos casos sean irrelevantes en algunos momentos de la prueba.

Primeramente se requiere de un dispositivo que permita visualizar los parámetros, no reste movilidad al supervisor y que no requiera interactuar físicamente con ninguna parte de la línea de proceso que incluye el equipo principal ni los transmisores en el momento de la prueba. Se toma esta medida por ser la primera vez que se pone operativo el equipo principal y por temas de seguridad al personal. Además el supervisor no debe tener la necesidad de portar dispositivos adicionales a los de su uso personal o

suministrado por la compañía a la que pertenece. Aplicando la tendencia BYOD que se está viendo en la actualidad en empleados de las compañías y como parte de una solución rentable se empleará el teléfono móvil del supervisor, o en forma general de un personal de comisionamiento.

Por otro lado se cuenta con transmisores que se encuentran en el equipo principal o en las líneas de proceso antes o después del mismo enviando las señales de los parámetros de proceso a la sala de control y no al supervisor del comisionamiento. Para ello se requiere el uso de un dispositivo que no sólo recoja las señales de 4-20mA emitida por los cables del transmisor sino también que éstas sean enviadas en forma inalámbrica al supervisor. Esto se puede lograr con el uso de un dispositivo instalado en serie en el transmisor.

Del análisis se desprende como principales puntos el uso de dos dispositivos:

- Teléfono móvil: A ser usado por el supervisor.
- Dispositivos en serie: A conectarse con el transmisor de proceso en forma serial y que se ubica cercano al mismo.

## 3.3 Análisis del hardware

### 3.3.1 Teléfono móvil

Las especificaciones técnicas mínimas que debe tener son:

- Comunicación inalámbrica para recibir las lecturas de los parámetros de proceso.
- Contar con una pantalla que permita ver las lecturas y como opción ver los gráficos de tendencias del comportamiento del parámetro de proceso.

Para detallar más el tipo de teléfono móvil que requiere la solución se deben tener en cuenta dos partes, la parte del hardware correspondiente a la tecnología de comunicación a usar, y en la parte del software un programa a instalar en el teléfono móvil.

Como se indicó, el teléfono a utilizar puede ser el de uso personal o suministrado por la compañía, es decir en este punto no involucra gastos adicionales para ninguna de las dos partes. El tipo de tecnología que se puede emplear para la comunicación inalámbrica debe ser el de menor costo. Se resalta que esta tecnología es incluida en las especificaciones técnicas del dispositivo en serie para que se dé la comunicación inalámbrica.

Para el caso de las tecnologías presentes en una planta industrial existen varias alternativas de comunicación, entre ellas se analiza las principales.

**TABLA N° 3.1** Comparación entre tecnologías de comunicación inalámbrica en plantas industriales [Elaboración propia].

| Tecnologia de comunicacion | Ventajas | Desventajas | Costo Referencial |
|---|---|---|---|
| **Redes de comunicacion para telefonia movil (GSM, GPRS, HSDPA entre otras)(*)** | Buena velocidad de trasnmision | Dependencia de la calidad del servicio, especificamente la continuidad del servicio. | Costo cero en la implementacion. Pero, el uso para fines que no son propios del contrato de servicio requerira una licencia y un pago mensual. |
| **Wifi/Wlan** | Alta velocidad de transmision. | Alto consumo de energia respecto a las demas tecnologias. Requiere equipos adicionales para habilitar la red. | Alto costo en la implementacion por requerir equipos adicionales para tener una red. Al usar la banda ISM( ), no involucra pago alguno por su uso. |
| **WirelessHart** | Permite diagnostico en linea tanto de la comunicacion como del transmisor en si. | Menor consumo de energia que su predecesor. Requiere equipos adicionales para instalar la red.<br>Baja velocidad de transmision. | Alto costo en la implementacion por requerir equipos adicionales para tener una red. Al usar la banda ISM( ), no involucra pago alguno. |
| **Bluetooth** | No requiere equipos adicionales aparte del transmisor y receptor.<br>Disponibilidad de esta tecnologia en la mayoria de telefonos moviles.<br>Bajo consumo de energia | No tiene proceso de diagnostico. Baja velocidad de transmision. | Costo cero en la implementacion. Costo cero en el uso de la banda por ser ISM. |

(*) GSM (Global System for Mobile), GPRS(General Packet Radio Service), HSDPA (High Speed Downlink Packet Access).

La Tabla 3.1 muestra las principales características de las tecnologías que actualmente se están aplicando en las plantas industriales. Realizando una evaluación técnica y económica, considerando solamente lo requerido para la solución, la mejor opción es la tecnología Bluetooth. Cabe indicar que no se requiere mucha velocidad de transferencia ya que son datos específicos que serán transmitidos desde dispositivo en serie al teléfono móvil, es por este motivo que la velocidad no presenta mucha relevancia para los requerimientos del presente trabajo.

### 3.3.2 Dispositivo en serie:

Las especificaciones técnicas mínimas que deberán tener son:

- Entrada de señal de 4-20mA.
- Modulo interno de ADC (Analog to Digital Converter – Convertidor análogo a digital).
- Contar con un módulo para transmisión de la señal Bluetooth.

- Conector para la instalación de una antena externa.
- Bajo consumo de energía.
- Compacto.

Es requerido que se pueda instalar una antena externa. El dispositivo se encontrará dentro del transmisor y para captar la señal Bluetooth la antena deberá estar expuesto al medio ambiente. Por eso se requiere que el dispositivo solo tenga el conector. La instalación de este dispositivo no tiene que afectar al transmisor por lo que también se requiere que consuma la menor cantidad de energía posible que está disponible y que no le reste espacio estando dentro del alojamiento del transmisor.

## 3.4 Análisis del software

Se tiene definido los dispositivos que se han de utilizar para la solución del problema. En el dispositivo en serie solamente se debe configurar para que después de recibida la señal de 4-20mA transmita su conversión equivalente vía radiofrecuencia. En el caso del teléfono móvil es distinto ya que requiere una presentación de la lectura de parámetros en la pantalla tan pronto son recibidas. Esto significa el desarrollo de una aplicación en la plataforma del teléfono móvil, a este tipo de aplicaciones se les llama APP (aplicación o programa para equipo móvil).

Para definir la plataforma se debe ver algunas graficas estadísticas respecto al porcentaje de preferencia de las diferentes marcas en el campo empresarial. Este dato es un indicador de la preferencia del empleado o del empleador. Para el análisis de las alternativas se toma en cuenta las características que se adelantaron respecto al teléfono móvil a usar en este informe.

Basado en un estudio de mercado realizado por International Data Corporation (IDC), las plataformas de mayor presencia en el mercado empresarial son Blackberry, iOS y Android. Tomando como referencia este estudio se podría considerar esta misma preferencia para los empleados locales [44].

Como se indicó en el Capítulo I "Marco Teórico Conceptual", cada una de estas plataformas tienen sus propios lenguajes de programación. El objetivo en este punto es usar un solo código fuente para estas tres diferentes plataformas.

Debido a las diferencias entre las plataformas una opción es el uso de un VM y/o como alternativa la conversión del archivo fuente del app.

Con la finalidad reducir el capital de inversión se opta por seleccionar el lenguaje Java para la generación del archivo fuente debido a que esta plataforma es código abierto, gratuita y con mayor soporte en el desarrollo de apps. Específicamente la plataforma del Java serie el J2ME por tener como dispositivo anfitrión un teléfono móvil.

**TABLA N° 3.2** Comparación entre plataformas de teléfonos móviles referente para instalación de una aplicación Java [Elaboración propia].

| Plataforma | Extension del App | Virtual Machine | Conversion de archivos Jar |
|---|---|---|---|
| iOS | *.ipa , *.m | No viene con VM. | Existe programa gratuito ( *.jar a *.class y luego a *.m) |
| Android | *.apk (empaquetado de archivos | DVM (Dalvik Virtual Machine) | Existen programas gratuitos (*.jar a *.dex y luego a *.apk) |
| Blackberry | *.alx | KVM (Kilo Virtual Machine) | No requerido al tener instalado el KVM (). |

La Tabla N° 3.2 muestra las características principales para la instalación de una aplicación Java en la plataforma correspondiente. Con la tabla se comprueba que es posible la instalación de una app en cualquiera de las tres plataformas mencionadas.

Dentro de los perfiles existentes, el más común que se encuentra en los teléfonos móviles es el SPP. Este perfil trabaja bajo el protocolo RFCOMM y está especificado en el JABWT (JSR-82) [45]. Por lo indicado será el perfil a utilizar para el presente informe.

## 3.5 Equipamiento

Por lo desarrollado anteriormente, las características requeridas serán las que se muestran en la Tabla N° 3.3.

**TABLA N° 3.3** Especificaciones técnicas en hardware y software requerido para el trabajo. [Elaboración propia].

| Equipos | Especificaciones técnicas | Descripción |
|---|---|---|
| Hardware | | |
| Teléfono móvil | Tecnología de comunicación | Bluetooth |
| Dispositivo en Serie | Señal de entrada | 4-20mA (*) |
| | Tecnología de comunicación | Bluetooth |
| | Montaje de antena | Externo |
| | Fuente de alimentación | 18 - 32Vdc (**) |
| | Consumo de energia | Es mas bajo posible |
| | Dimension | Compacto |
| Software | | |
| Teléfono móvil | Plataforma | BlackBerry, iOS o Android. |
| | App | Bajo plataforma J2ME |
| Dispositivo en Serie | | Solamente requiere configuración. |

(*) Corriente con que trabajan normalmente los transmisores industriales.
(**) Voltaje con que trabajan normalmente los transmisores industriales.

A partir de la Tabla N° 3.3 se detalla la mejor opción para la solución.

### 3.5.1 Parte del Hardware

#### a. Teléfono móvil

La finalidad es tener la mayor cantidad de teléfonos móviles disponibles en una planta industrial, y se logra con la menor cantidad de especificaciones técnicas posibles. Solamente se solicita que presente la tecnología Bluetooth y en la actualidad casi la mayoría lo tiene.

#### b. Dispositivo en serie

Se tiene en el mercado diferentes marcas.

**TABLA N° 3.4** Comparación entre alternativas para el dispositivo en serie [Elaboración propia].

| Marca | Modelo | Especificacion tecnicas | Aplicacion industrial |
|---|---|---|---|
| **Bluegiga** | WT-32 | Cumple con las especificaciones tecncas minimas.<br>No tiene entrada 4-20mA directa, requerira de un convertidor.<br>Tiene conector tipo UFL para la antena. | Si cumple |
| **Roving Networks** | RN-24-E | Cumple con las especificaciones tecncas minimas.<br>No tiene entrada 4-20mA directa, requerira de un convertidor.<br>Tiene conector coaxial para la antena tipo SMA | Si cumple |

En la Tabla N° 3.4 ambos módulos cumplen con lo requerido. El dispositivo seleccionado es de la empresa Roving Networks debido a que incluye un conector SMA el cual es más manejable que los conectores tipo UFL que son muy pequeños y frágiles.

Se deben considerar las siguientes partes adicionales:

### b.1 Antena del dispositivo

El conector SMA hembra del dispositivo permite utilizar un cable coaxial con el conector macho para extender la antena hasta el exterior, el otro extremo que no incorpora conector, la punta solo esta protegida con una manga termo contraíble. Para este extremo se instala una caja condulet por donde sale el cable coaxial usado como antena y para fijarlo se rellena con silicona. La manga termo contraíble también se usa en la conexión SMA macho-hembra de la antena.

### b.2 Fuente de alimentación

Para no afectar las funciones del transmisor, el dispositivo en serie incluye una fuente de alimentación externa que puede ser una batería de teléfono celular para alargar el tiempo su tiempo de uso y permita una recarga del dispositivo. Recordar que los dispositivos en serie solamente serán usados en la etapa de comisionamiento, es decir se instala temporalmente.

### b.3 Acondicionamiento de la señal

La señal del transmisor es de 4-20mA y a pesar de que existan otras cargas en serie el transmisor mantiene esta corriente. Por lo tanto para recoger la señal de corriente se utilizara una resistencia de precisión. La señal de entrada es en voltaje y tiene rango de 0-1.8Vdc. El valor de la resistencia requerida es de 90 ohms.

### b.4 Alojamiento del dispositivo

Todo el circuito se acondiciona en una tarjeta. A excepción de los conectores para la batería, antena y la entrada de señal, toda la tarjeta se recubre con una resina epoxi haciéndola aislante, compacta y que al estar dentro del alojamiento del transmisor no genere cortocircuito al contacto con la electrónica del mismo.

### 3.5.2 Parte del Software

El desarrollo del app está en la plataforma J2ME cuya elección fue sustentada en los párrafos previos. Aplicando el método "Top-down" al app a elaborar, la Fig. 3.1 muestra el diagrama de flujo correspondiente con las diferentes pantallas que se ve en el teléfono móvil.

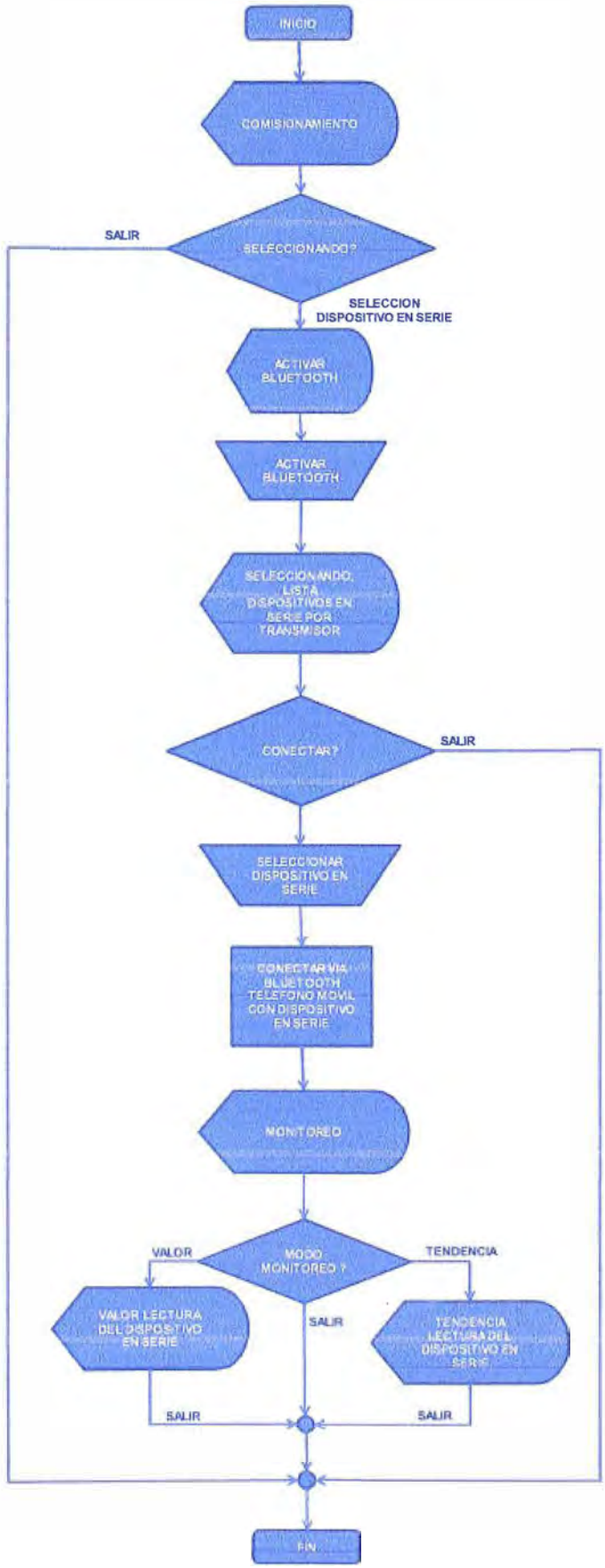



**Fig.3.1** Diagrama de flujo del app [Elaboración propia].

Dentro de las principales funcionalidades para la programación en plataforma J2ME, el app a implementar utiliza las siguientes:

- **Especificaciones del JSR-82**

Con el paquete "javax.bluetooth" que para el caso del app cliente avise de posibles errores en la conexión por la tecnología Bluetooth entre el teléfono móvil y el dispositivo en serie.

- **Técnica multithreads (multihilos)**

El establecimiento de la conexión entre ambos equipos es de lapsos que detendrán el avance del app. Para ello se hace uso de la técnica multithreads permitiendo que el proceso principal de la app continúe mientras otro proceso (thread) en paralelo concluya estableciendo la conexión vía Bluetooth entre el teléfono móvil y el dispositivo en serie.

- **Clase Canvas**

Las tendencias, que son representaciones gráficas de las lecturas de un determinado parámetro de proceso, requieren el manejo a nivel de pixeles de la pantalla del teléfono móvil. La clase Canvas permiten realizar esta función creando interfaces llamadas de bajo nivel por el manejo de los gráficos a partir de los pixeles.

**a. Elementos del Diagrama de flujo**

Entre los elementos que conforman el diagrama de flujo se encuentran:

**a.1 Inicio (Terminal)**

Elemento indicador del inicio del app. Esta app corresponde al lado cliente de una comunicación.

El app esta conformado principalmente por una subclase de Midlet que permite ser ejecutable en el teléfono móvil.

**a.2 Comisionamiento (Pantalla)**

A través del objeto List que representa la pantalla "Comisionamiento" muestra dos opciones considerando el comando de salida del app.

**a.3 Seleccionando? (Decisión)**

Conformada por dos opciones que son comandos agregados al objeto List "Comisionamiento". La opción "Seleccionando" da pase a la lista de dispositivos en serie.

**a.4 Activar Bluetooth (Pantalla)**

Antes de pasar al formulario "Seleccionando", el app lanzará una alerta de aviso al usuario que debe de activar la Tecnología Bluetooth del teléfono móvil. Se da por aceptada la alerta al presionar el botón correspondiente del teléfono móvil.

**a.5 Seleccionando (Pantalla)**

Usando un objeto de tipo Form mostrará una lista con todos los dispositivos en serie. Para la lista se usa el objeto ChoiceGroup que dentro de sus elementos tiene los

tags (etiquetas) de los transmisores en donde están instalados los dispositivos en serie. Cada uno de los elementos del objeto ChoiceGroup se relaciona con la asignación de la dirección Bluetooth que todo equipo presenta esta tecnología, así también el número del Server Channel (Canal del servidor), que en su mayoría es de valor "1". Ambas asignaciones evita el uso de los servicios Device Discovery (Descubrimiento del dispositivo) y Service Discovery (Descubrimiento del servicio) que son propios del JABWT y que pueden alargar el proceso de ejecución del app. Estas asignaciones junto con la sintaxis para uso del perfil SPP (btspp://"Dirección Bluetooth":"Canal del servidor"), forma los datos para el objeto String conocido como Connection String (String o cadena de texto para conexión) que es un paso para el pedido de comunicación al dispositivo en serie.

**a.6 Seleccionar dispositivo en serie (Operación manual)**

Esta operación la maneja el usuario y se acepta la selección con el uso del comando respectivo representado por un botón en la pantalla "Seleccionando". En este punto se genera el objeto Connection String.

**a.7 Conectar vía Bluetooth (Proceso)**

Este proceso se inicia con la activación del botón de la pantalla "Seleccionando". A partir del objeto Connection String, con el uso del método Open del objeto Connector y conversión de objeto a StreamConnection se ejecuta el pedido de servicio o solicitud de conexión al dispositivo en serie. Este proceso debido a que no tiene tiempo determinado por la dependencia de respuesta del dispositivo remoto es que se hace uso de la técnica multithreading.

**a.8 Monitoreo (Pantalla)**

Representado en la programación por un objeto List, mostrará dos opciones para continuar con la ejecución del app.

**a.9 Modo Monitoreo? (Decisión)**

Se tienen dos opciones para la forma que se muestran las lecturas. Estas opciones que son comandos derivan el avance del app hacia dos pantallas.

**a.10 Valor, lectura del dispositivo en serie (Pantalla)**

Representado en la programación por el objeto Form, mostrará la lectura del parámetro de proceso en dígitos.

**a.11 Tendencia, lectura del dispositivo en serie (Pantalla)**

A partir de la opción escogida del elemento Decisión "Modo Monitoreo?", se pasa a la generación del objeto de clase Canvas que dibuja en la pantalla la representación gráfica de las lecturas del parámetro de proceso. Para este punto se implementa una subclase de Canvas el cual se llama desde el app. Cabe indicar el uso del paquete "java.util" donde se especifican las clases Timer y TimerTask que permite actualizar la

tendencia mostrada.

**a.12 Fin (Terminal)**

Elemento indicador del término del app.

**b. Bosquejo de las pantallas**

La aplicación a desarrollar está conformada por:

**b.1 Pantalla principal (Titulo: COMISIONAMIENTO)**

Donde se escoge que procedimiento seguir:

- Selección del transmisor a monitorear, o,
- Monitorear la señal del transmisor.

Entenderse como señal del transmisor al que es enviado por el dispositivo en serie vía Bluetooth. Así también para el procedimiento de selección.

**b.2 Pantalla para selección del transmisor (Titulo: SELECCIONANDO)**

Muestra una lista de transmisores que están instalados los dispositivos en serie. Se marca el que se desea monitorear.

**b.3 Pantalla para selección del modo de presentación (Titulo: MONITOREO)**

Para escoger el modo de mostrar la lectura de los parámetros, por valores o tendencias.

**b.4 Pantalla de lectura por valores (Titulo: VALOR)**

Muestra las lecturas de los parámetros en dígitos.

**b.5 Pantalla de lectura por tendencias (Titulo: TENDENCIA)**

Muestra las lecturas de los parámetros por tendencias.

**3.6 Capacitación del personal de comisionamiento**

El personal no requiere de capacitación ya que el manejo de la aplicación fue diseñado para que su manejo sea en forma intuitiva. De requerir esta capacitación ésta es de pocos minutos.

**3.7 Recursos humanos**

Entre los principales se tiene:

**3.7.1 Preparación del Hardware**

**a. Teléfono Móvil**

No requiere mayor acondicionamiento, solamente que esté habilitada la comunicación Bluetooth que fácilmente lo puede realizar el usuario del teléfono.

**b. Dispositivo en serie**

Teniendo el dispositivo a la mano se tendría que acondicionar con lo indicado en el párrafo 3.5.1 Parte del Hardware antes de instalarlo dentro del transmisor. Un técnico electrónico podría realizar el trabajo, tanto el armado del dispositivo como su instalación en campo.

### 3.7.2 Preparación del Software

**a. Programación del app**

Se requiere de un personal con conocimiento de Programación Orientada a Objetos (POO) y programación de aplicaciones basado en tecnología Bluetooth con lenguaje Java. La actividad del personal concluye con la transferencia de datos entre el teléfono móvil y el dispositivo en serie. Las características del trabajo describen a las labores de un programador de sistemas con conocimiento de electrónica.

**b. Configuración del dispositivo en serie**

De las especificaciones técnicas de la tarjeta seleccionada (Marca: Roving Networks, Modelo: RN-24-E), tampoco se requiere de una configuración. Por lo tanto en esta parte no se requiere de personal especializado.

# CAPÍTULO IV
# ANÁLISIS Y PRESENTACIÓN DE RESULTADOS

## 4.1 Introducción

En este capítulo se simula la comunicación entre el teléfono móvil y el dispositivo en serie con el uso de equipos que cubran la mayoría de las especificaciones técnicas requeridas e indicadas en el Capítulo III "Metodologla para la solución del problema"

## 4.2 Equipamiento

### 4.2.1 Teléfono móvil

El requerimiento es mínimo por lo que se selecciona el siguiente teléfono:

Marca: Blackberry

Modelo: 9700 o 9800.

Entre las características principales incluye:

- Tecnología Bluetooth
- Configuración / perfil de la plataforma J2ME: CLDC v.1.1 / MIDP v.2.1
- SPP

Se resalta el perfil SPP que contiene el servicio Serial Port porque a través del protocolo RFCOMM que usa se establecerá la comunicación.

### 4.2.2 Dispositivo en serie

Se selecciona un dispositivo que tenga casi todas las características principales del requerido.

**a. Dispositivo requerido**

Nombre del dispositivo: Class 1 Bluetooth Super Module

Marca: Roving Network

Modelo: RN-24-E

Entre las caracterlsticas principales incluye:

- Tecnología Bluetooth.
- SPP.
- Rango de señal de entrada analógica: 0-1.8Vdc.
- Fuente de alimentación: 4-24Vdc.
- Consumo de energía: 65mA (típico trabajando como transmisor).

- Conector SMA para antena externa.
- Dimensiones: 4.4 x 2.8 x 0.5 cm (compacto).

**b. Dispositivo reemplazo**

El equipo que realiza las funciones de dispositivo requerido sólo por fines de prueba es:

Equipo: Laptop con Tecnología Bluetooth

Marca: HP Pavilion

Modelo: dv5-2247la

Entre las características principales incluye:

- Tecnología Bluetooth.
- SPP.
- Fuente de alimentación: cargador de batería autovoltaje.
- Antena ubicada internamente.

Como se observa tiene cubierta las principales características requeridas por el dispositivo en serie.

### 4.3 Programación del app

La programación se ha realizado con los siguientes programas softwares:

- Netbeans IDE 7.2.
- Sun Java Wireless Toolkit 2.5.2.

Configuración y perfil seteados en el app:

- CLDC v.1.0
- MIDP v. 1.0

Se ha seleccionado estas versiones para evitar problemas con la compatibilidad en los teléfonos móviles y así mantener la gama de alternativas.

#### 4.3.1 Código fuente del app

Compuesta por cuatro subclases:

- BluetoothMain: Subclase de MIDlet del paquete "javax.microedition.midlet".
- MidletTitu: Subclase de BluetoothMain.
- TareaSeg: Subclase de TimerTask del paquete "java.util".
- TendCanvas: Subclase de Canvas del paquete "javax.microedition.lcdui".

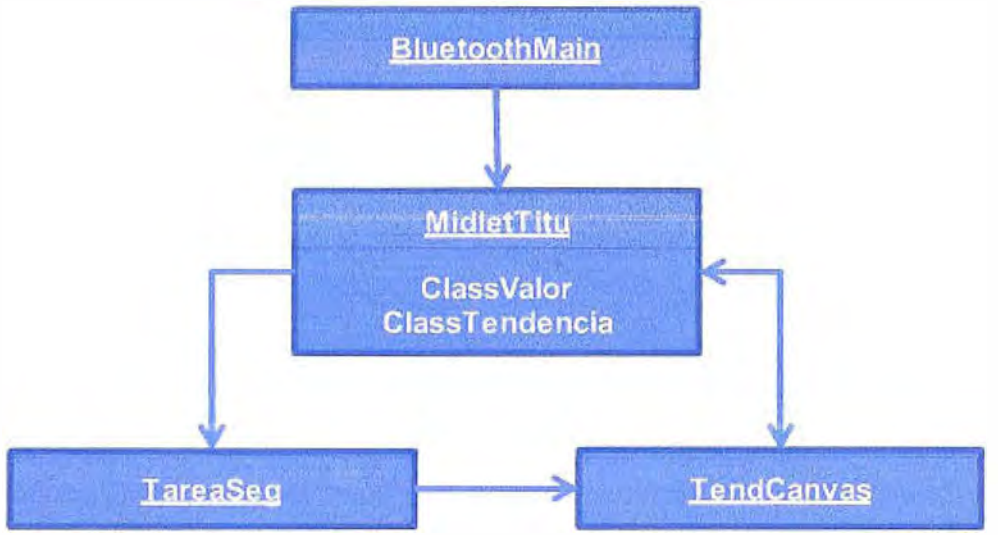



**Fig.4.1** Diagrama de bloques del app [Elaboración propia].

La Fig.4.1 muestra las clases utilizadas y las fechas indican la generación de cada objeto a partir de la clase respectiva.

Para la generación del app se tiene que usar una subclase proveniente de la clase Midlet. La subclase BluetoothMain extendida de la clase Midlet es la base para la subclase principal MidletTitu. Pero además debido a que involucrará comunicación entre equipos se aplica la técnica multithreading a partir de la interface Runnable implementada en las clases respectivas. La interface Runnable aumenta la rapidez de procesamiento del app. Este app considera el dibujo de gráficos (Tendencias) de las lecturas que se obtengan. Debido a que en los APIs de alto nivel para dibujar en la pantalla no tiene las herramientas requeridas es que se usa los APIs de bajo nivel en donde por medio de la clase Canvas y de sus métodos respectivos se puede dibujar las tendencias. Las tendencias se deben de actualizar una vez se reciba los datos enviados por el dispositivo en serie, es decir no requiere intervención del usuario. Esto último es posible con el uso de la subclase TareaSeg de la clase TimerTask. Esta clase permite realizar tareas periódicas que en este caso se usa para actualizar la tendencia conforme se reciban los datos vía la tecnología Bluetooth.

Ver Anexo (A).

**a. BluetoothMain**

Se importan los paquetes requeridos entre los cuales destaca "javax.bluetooth" que avisará de posibles errores en la conexión vía la tecnología Bluetooth.

Código en referencia:

```
import javax.microedition.midlet.*;
```

```
import javax.microedition.lcdui.*;
import java.lang.*;
import java.io.*;
import javax.microedition.io.*;
import javax.bluetooth.*;
```

Seguidamente se muestra la estructura que requiere ser implementado en el BluetoothMain por ser subclase de Midlet, entre ellos los métodos startApp( ), pauseApp( ) y destroyApp( ) . Esta clase incluye la interface CommandListener para detectar los eventos en el uso de los comandos y que incluye el método commandAction( ). Además está la interface Runnable para el uso de la técnica multithreading que incluye el método run( ).

Código en referencia:

```
public class BluetoothMain extends MIDlet implements CommandListener, Runnable {
    public void startApp() {
    }
    public void pauseApp() {
    }
    public void destroyApp(boolean unconditional) {
    }
    public void commandAction(Command c, Displayable d){
    }
    public void run(){
    }
}
```

**b. MidletTitu**

Esta es la clase principal. Como subclase de BluetoothMain incluye la interface Runnable. Además se agrega el paquete "java.util" para poder utilizar la clase Timer.

Código en referencia:

```
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;
import java.lang.*;
import java.io.*;
import javax.microedition.io.*;
import javax.bluetooth.*;
import java.util.*;
```

Como se indicó el MidletTitu es una subclase de BluetoothMain y para detectar el uso de los comandos se usa la interface CommandListener. Seguidamente se declara los objetos y variables que se usa.

Código en referencia:

```
    private Display display;
    private List COMISIONAMIENTO;
    private Form SELECCIONANDO;
```

```
private List MONITOREO;
private Form VALOR;
private Alert ACTIVARBLUETOOTH;
private Command cSalir;
private ChoiceGroup listDisp;
private TextField tIngrServ;
private Command cConectar;
private int iCanal;
public String sPara,sParaUnit;
private String coneString;
private Command cRegresar;
private StreamConnection coneConector;
public DataInputStream disDatos;
private byte[] bValor;
public int iValor;
public int iTendencia;
private TendCanvas TENDENCIA;
private Timer Secuenciador;
private TareaSeg tareaseg;
private Thread thValor;
private Thread thTendencia;
```

En el constructor del MidletTitu se empieza a crear los objetos con sus parámetros respectivos.

Para operar con la pantalla del teléfono móvil se debe tener un objeto del tipo Display con referencia a la pantalla de la clase MidletTitu.

También se construye listas, formularios, alertas, comandos y todos los demás objetos a utilizar en la Clase MidletTitu. Principalmente incluye: objeto list "COMISIONAMIENTO", objeto alert "ACTIVARBLUETOOTH", objeto command "cSalir", objeto form "SELECCIONANDO", objeto list "MONITOREO", objeto form "VALOR", objeto canvas "TENDENCIA" y objeto timer "Secuenciador". Todo ello encerrado en un try-catch que avisa cuando ocurra una anomalía en la ejecución del app. Tanto el objeto form como list ocupan toda la pantalla del teléfono móvil. Pero en el caso del objeto form, las opciones para escoger a partir de la pantalla se encuentran ocultas y se muestran después de presionar un botón. Para el objeto list como en "COMISIONAMIENTO" y "MONITOREO", las opciones ya se encuentran en la pantalla facilitando su selección. El objeto ChoiceGroup que se agrega al objeto form "SELECCIONANDO" permite una única selección de tag del transmisor y se muestra en el misma pantalla sin necesidad de estar trabajando con otra adicional.

Cabe indicar que existe un objeto TextField para tipear el Canal del Servidor. Este objeto no sería necesario en el caso real porque este dato no cambia, pero si para la demostración del app cliente ya que utilizara como dispositivo en serie un equipo reemplazo.

Código en referencia:

```
public MidletTitu(){
 try{
 display=Display.getDisplay(this);
 COMISIONAMIENTO = new List("COMISIONAMIENTO",List.IMPLICIT);
 COMISIONAMIENTO.insert(0,"SELECCIONANDO", null);
 cSalir=new Command("SALIR",Command.EXIT,1);
 COMISIONAMIENTO.addCommand(cSalir);
 COMISIONAMIENTO.setCommandListener(this);
 ACTIVARBLUETOOTH=new Alert("ACTIVAR BLUETOOTH","Presionar boton de
reconocimiento \"Done\"",null,AlertType.WARNING);
 ACTIVARBLUETOOTH.setTimeout(Alert.FOREVER);

//  Form: SELECCIONANDO

 listDisp=new ChoiceGroup("TRANSMISORES",Choice.EXCLUSIVE);
 listDisp.append("PT100001", null);
 listDisp.append("VT100001", null);
 listDisp.append("TT100001", null);
 listDisp.append("PT100002", null);
 listDisp.append("VT100002", null);
 listDisp.append("TT100002", null);
 listDisp.setSelectedFlags(new boolean[]{false, false, false, false, false, false});
 tIngrServ=new TextField("INGRESAR CANAL DEL SERVIDOR:
","",2,TextField.NUMERIC);
 cConectar=new Command("CONECTAR",Command.OK,1);
 SELECCIONANDO=new Form("SELECCIONANDO", new Item[]{listDisp,tIngrServ});
 SELECCIONANDO.addCommand(cSalir);
 SELECCIONANDO.addCommand(cConectar);
 SELECCIONANDO.setCommandListener(this);

 //  List: MONITOREO
 MONITOREO=new List("MONITOREO",List.IMPLICIT);
 MONITOREO.insert(0,"TENDENCIA", null);
 MONITOREO.insert(0,"VALOR", null);
 MONITOREO.addCommand(cSalir);
 MONITOREO.setCommandListener(this);

//  Form: VALOR
 VALOR=new Form("VALOR");
 VALOR.addCommand(cSalir);
 VALOR.setCommandListener(this);

//  TendCanvas: TENDENCIA
 TENDENCIA=new TendCanvas(this);

//  Timer: Secuenciador
 Secuenciador=new Timer();
 tareaseg=new TareaSeg(TENDENCIA);
 }
 catch (Exception e){
  System.out.println("EXCEPCION:  " + e);
 }
```

```
}
```

Como parte de la estructura del Midlet se implementan los métodos startApp(), pauseApp() y destroyApp(). Para el método startApp(), se muestra la pantalla del objeto list "COMISIONAMIENTO" como inicio del app en el teléfono móvil. En el método destroyApp() se cierran los objetos utilizados en la conexión Bluetooth.

Código en referencia:

```
public void startApp() {
   display.setCurrent(COMISIONAMIENTO);
}
public void pauseApp() {
}
public void destroyApp(boolean unconditional) {
try{
   disDatos.close();
   coneConector.close();
}
catch(Exception e){
   System.out.println("Exception es:"+e);
}
}
```

Para el método commandAction() que se ejecuta ante el accionamiento de un comando. Estando en la pantalla del objeto list "COMISIONAMIENTO" se tienen dos opciones: el objeto command "cSalir" u objeto form "SELECCIONANDO", este último está unido a un objeto alert "ACTIVARBLUETOOTH" que ocupa toda la pantalla y se muestra antes que el objeto form. El objeto command "cSalir" con el llamado a los métodos destroyApp() y notifyDestroyed() cierra el app.

Código en referencia:

```
public void commandAction(Command c,Displayable d){
   if (d==COMISIONAMIENTO){
      if (c==cSalir){
         destroyApp(true);
         notifyDestroyed();
      }
      if (c==COMISIONAMIENTO.SELECT_COMMAND){
         display.setCurrent(ACTIVARBLUETOOTH, SELECCIONANDO);
      }
   }
```

De encontrarse en la pantalla del objeto form "SELECCIONANDO" se tiene la lista de los transmisores en donde se escoge sólo uno para comunicarse. El objeto command cConectar primeramente guarda el Canal de Servidor tipeado, luego evalua que etiqueta (tag) de transmisor fue seleccionado. Así el número de Bluetooth que cada equipo cuenta con esta tecnología se almacena para ser usado en toda la ejecución del app. Con el uso

de la función switch() se almacena la sintaxis correspondiente al Connection String que incluye el número de Bluetooth, el Canal del Servidor y la unidad de medida utilizada.

Código en referencia:

```
if (d==SELECCIONANDO){
   if (c==cConectar){
      iCanal=Integer.parseInt(tIngrServ.getString());
      switch(listDisp.getSelectedIndex()){
          case 0:{
              sPara = listDisp.getString(0);
              sParaUnit="psi";
              coneString="btspp://184617e3db11"+":"+iCanal;
//              PT100001=TABLET
              break;}
          case 1:{
              sPara = listDisp.getString(1);
              sParaUnit="mm/s";
              coneString="btspp://f40b93b77c22"+":"+iCanal;
//              VT100001=9700
              break;}
         case 2: {
              sPara = listDisp.getString(2);
              sParaUnit = "ºC";
              coneString = "btspp://cc52af019c1e"+":"+iCanal;
  //              TT100001=LAPTOP
                break;}
          case 3:{
              sPara = listDisp.getString(3);
              sParaUnit="psi";
              coneString="btspp://0018003603f4"+":"+iCanal;
//              PT100002=TECLADO
              break;}
          case 4:{
              sPara = listDisp.getString(4);
              sParaUnit="mm/s";
              coneString="btspp://cc55ad26865b"+":"+iCanal;
//              VT100002=9800
              break;}
         case 5: {
              sPara = listDisp.getString(5);
              sParaUnit = "ºC";
              coneString = "btspp://"+":"+iCanal;
  //              TT100002=
              break;}
      }
```

El establecimiento de conexión con el dispositivo en serie seleccionado es un paso que toma un tiempo indeterminado. Para acelerar la conexión se usa la técnica multithreading a través de la creación de un objeto tipo Thread propio del MidletTitu y

seguidamente se iniciará. A la par se muestra la pantalla del objeto list "MONITOREO" en donde se tiene tres opciones: el objeto command "cSalir" (para finalizar el app), el objeto command "VALOR" (para ir a la pantalla del objeto form "VALOR") y el objeto command "TENDENCIA" (para ir a la pantalla del objeto Canvas "TENDENCIA").

Para el objeto command "VALOR", previamente se genera e inicia el objeto thread "thValor" para la ejecución del objeto de clase ClassValor que tiene interface de nombre Runnable.

Para el objeto command "TENDENCIA", previamente se genera e inicia el objeto thread "thTendencia" para la ejecución del objeto de clase ClassTendencia que tiene interface Runnable.

Código en referencia:

```
//          CONECTARSE CON DISPOSITIVO EN SERIE
//          CREAR EL DATAINPUTSTREAM
                new Thread(this).start();
             }
           display.setCurrent(MONITOREO);
           if (c==cSalir){
            destroyApp(true);
            notifyDestroyed();
           }
           }
       if (d==MONITOREO){
           if (c==List.SELECT_COMMAND){
              switch(MONITOREO.getSelectedIndex()){
                    case 0:{
                    //      IR A LA PANTALLA DE VALOR
                    thValor=new Thread(new ClassValor(disDatos));
                    thValor.start();
                    break;
                    }
                    case 1:{
                     //      IR A LA PANTALLA DE TENDENCIA
                    Secuenciador.schedule(tareaseg, 0, 1000);
                    thTendencia=new Thread(new ClassTendencia(disDatos));
                    thTendencia.start();
                    break;
                    }
              }
           }
           if (c==cSalir){
              destroyApp(true);
              notifyDestroyed();
           }
       }
```

Dentro del método commandAction() también se tiene el objeto command "cSalir" (cerrar el app) tanto para el objeto form "VALOR" como el objeto Canvas "SALIR".

Código en referencia:

```
    if (d==VALOR){
       if (c==cSalir){
          destroyApp(true);
          notifyDestroyed();
       }
    }
    if (d==TENDENCIA){
       if (c==cSalir){
          destroyApp(true);
          notifyDestroyed();
       }
    }
}
```

Este es el método run() propio del MidletTitu para el establecimiento de la conexión. En este punto se utiliza el objeto ConnectionString "coneString" obtenido anteriormente en el método open() del objeto Connector, y para obtener el objeto StreamConnection se utiliza la conversión de objeto (StreamConnection) obteniéndose el "coneConector". Seguidamente se genera el objeto DataInputStream "disDatos" a partir del objeto "coneConector" y su método openInputStream(). El uso del try-catch avisa de posibles problemas en la conexión.

Código en referencia:

```
public void run(){
   try{
      coneConector=(StreamConnection)Connector.open(coneString);
      disDatos = new DataInputStream(coneConector.openInputStream());
     }
   catch(Exception e){
      System.out.println("EXCEPCION ES:"+e);
      SELECCIONANDO.append("EXCEPCION ES:"+e);
      }
}
```

Debido a que se recibe los datos en bytes se tiene que convertir a decimal, para ello se utiliza el método metLeerDatos() que requiere como argumento el objeto DataInputStream "disDatos" creado anteriormente. En este método se lee los bytes enviados y los convierte a decimal. Por ejemplo para el carácter "4" le corresponde el valor decimal 52 que restándole 48 se obtiene el valor decimal "4". Este procedimiento se puede aplicar para todos los bytes recibidos entendiéndose como el primer byte el más significativo. El valor final es el dato de salida del método. Se está utilizando el try-catch para avisar si ocurre algún problema en la lectura de los bytes, de ser así el resultado del método envia el valor "77" indicando la existencia de una anomalía.

Código en referencia:

```
public int metLeerDatos(DataInputStream dis1){
   byte[] bmetLD=new byte[4];
   try{
     int iLong=dis1.read(bmetLD);

     int imetValor=0;
     for(int j=iLong;j>0;j--){
        bmetLD[j-1]-=48;
        for(int k=iLong-j;k>0;k--){
           bmetLD[j-1]*=10;
        }
        imetValor+=bmetLD[j-1];
     }
     return imetValor;
   }
   catch(Exception e){
       return 77;
   }
}
```

Dentro de la pantalla del objeto list "MONITOREO" se indica el objeto command "VALOR" el cual al ser activado se inicia el objeto thread "thValor" y se ejecuta el objeto ClassValor con interface Runnable. En esta clase con el uso del objeto DataInputStream "disDatos" y el método metLeerDatos() se obtiene el objeto String para mostrarlo en la pantalla del objeto form "VALOR" la lectura enviada por el dispositivo en serie seleccionado.

Código en referencia:

```
class ClassValor implements Runnable {
   private DataInputStream disClassValor;
   public ClassValor(DataInputStream disV){
      this.disClassValor=disV;
   }
   public void run(){
      display.setCurrent(VALOR);
      while (display.getCurrent()==VALOR) {
         iValor=metLeerDatos(disClassValor);
         iTendencia=iValor;
         if (VALOR.size()>0){
            VALOR.delete(0);
         }
         VALOR.append(sPara+"= "+iValor+sParaUnit);
      }
    }
}
```

Al igual que la clase anterior, la clase ClassTendencia con interface de nombre Runnable también utiliza el objeto DataInputStream "disDatos". En esta clase se llama al objeto de

la clase TendCanvas "TENDENCIA" para iniciar el dibujo del gráfico con las lecturas recibidas desde el dispositivo en serie.

Código en referencia:

```
class ClassTendencia implements Runnable {
   private DataInputStream disClassTendencia;
   public ClassTendencia(DataInputStream disT){
      this.disClassTendencia=disT;
      }

   public void run(){
     TENDENCIA.bBack=true;
     display.setCurrent(TENDENCIA);
     while (display.getCurrent()==TENDENCIA) {
        iTendencia=metLeerDatos(disClassTendencia);
        iValor=iTendencia;
     }
   }
}
```

El metodo cSalTendCanvas() se activará desde la clase TendCanvas "TENDENCIA" para cerrar el app.

Código en referencia:

```
public void cSalTendCanvas (){
   destroyApp(true);
   notifyDestroyed();
}
}
```

**c. TareaSeg**

Esta es la subclase de TimerTask y requiere del paquete "java.util" que se encuentra al inicio de su implementación. Esta clase permite actualizar con los datos enviados por el dispositivo en serie las variables del objeto TendCanvas "TENDENCIA". Luego se ejecuta el método repaint() propio del TendCanvas para agregar más líneas al gráfico mostrado en la pantalla.

Código en referencia:

```
import java.util.*;
class TareaSeg extends TimerTask {
   private TendCanvas tCanvas;
   public TareaSeg (TendCanvas ttTend){
      this.tCanvas=ttTend;
   }

   public final void run(){
      tCanvas.x0=tCanvas.x1;
      tCanvas.x1+=tCanvas.gwSeg;
      tCanvas.y0=tCanvas.y1;
```

```
        tCanvas.repaint();
    }
}
```

**d. TendCanvas**

Esta es subclase de Canvas. Para el uso de la clase Canvas requiere del paquete "javax.microedition.lcdui". Tiene la interface CommandListener para el uso de los comandos que se agreguen. En la primera parte se declarara las variables y objetos a utilizar. Se declara el objeto mTend de clase MidletTitu para el intercambio de datos entre los objetos de la clase MidletTitu y TendCanvas. El objeto command rMonitoreo permite ejecutar un método externo.

Código en referencia:

```
import javax.microedition.lcdui.*;
public class TendCanvas extends Canvas implements CommandListener {
    int gh;
    int gw;
    int x;
    int y;
    int xof=7;
    int yof=7;
    int xvar=2;
    int yvar=30;
    int ghGraf;
    int gwGraf;
    int xGraf;
    int yGraf;
    int iValorTend;
    int gwSeg=2;
    int xSeg;
    int ySeg;
    int x1;
    int y1;
    int x0;
    int y0;
    int nSeg;
    boolean bBack;
    private MidletTitu mTend;
    private Command rMonitoreo;
```

El constructor de la clase TendCanvas requiere como argumento el objeto MidletTitu de donde es llamado. Dentro de las primeras variables se tiene "gh" y "gw" que son las dimensiones en pixeles de la pantalla del teléfono móvil, mientras que "ghGraf" y "gwGraf" son las dimensiones en pixeles del área donde se dibuja el gráfico. "x0" e "y0" es el punto inicial de la línea a dibujar, mientras que "x1" e "y1" es el punto final de la línea correspondiente. Se agrega el objeto command "rMonitoreo" al objeto de clase. Todas estas sentencias son encerradas en la función try-catch para que avise de una

anomalía en la ejecución. También está el método estInicial() que establece los valores iniciales para dibujar el gráfico, es decir donde inicia el primer trazo en el gráfico.

Código en referencia:

```
public TendCanvas(MidletTitu mLocal) {
    this.mTend=mLocal;
    gh=this.getHeight();
    gw=this.getWidth();
    ghGraf=gh-yof-yvar;
    gwGraf=gw-xof-xvar;
    try {
        rMonitoreo=new Command("SALIR",Command.EXIT,1);
        this.addCommand(rMonitoreo);
        setCommandListener(this);
        bBack=true;
        }
    catch (Exception e) {
        e.printStackTrace();
    }
    estInicial();
}
```

Para iniciar el gráfico en la pantalla del objeto TendCanvas se usa el método paint() el cual después de la primera ejecución se llama por el método repaint(). La variable booleana "bBack" seteado como "true" en el objeto de clase MidletTitu en la clase ClassTendencia permite dibujar por una sola vez el fondo que presenta el gráfico, en este caso se dibujan los ejes x e y a través del método drawLine(). Seguido con el uso del método setClip() se selecciona una franja superior para mostrar tanto el tag del transmisor seleccionado como la lectura enviada por el dispositivo en serie. Finalmente, se vuelve a usar el método setClip() para seleccionar toda la pantalla y poder dibujar la línea correspondiente a la lectura recibida con el método drawLine().

Código en referencia:

```
public void paint(Graphics g) {
    if (bBack==true){
        g.setColor(255,255,255);
        g.fillRect(0, 0, gw, gh);
        g.setColor(0, 0, 0);
        g.drawLine(conx(0),cony(0),conx(gwGraf),cony(0));
        g.drawLine(conx(0),cony(0),conx(0),cony(ghGraf));
        bBack=false;
    }
    g.setClip(0,0,gw,yvar);
    g.setColor(255,255,255);
    g.fillRect(0, 0, gw, gh);
    g.setColor(0, 0, 0);
    g.drawString(mTend.sPara + "= "+mTend.iTendencia+mTend.sParaUnit, gw-1, 0,
Graphics.TOP | Graphics.RIGHT);
    g.setClip(0,0,gw,gh);
```

```
        g.drawLine(conx(x0),cony(y0),conx(x1),cony(mTend.iTendencia));
        y1=mTend.iTendencia;
    }
```

Los métodos siguientes se tienen que implementar por ser TendCanvas una subclase de Canvas.

Código en referencia:

```
    protected void keyPressed(int keyCode) {
    }

    /**
     *
     * Llamado cuando una tecla es liberada
     */
    protected void keyReleased(int keyCode) {
    }

    /**
     * Llamado cuando una tecla es repetida
     */
    protected void keyRepeated(int keyCode) {
    }

    /**
     * Llamado cuando el apuntador es arrastrado.
     */
    protected void pointerDragged(int x, int y) {
    }

    /**
     * Llamado cuando el apuntador es presionado.
     */
    protected void pointerPressed(int x, int y) {
    }

    /**
     * Llamado cuando el apuntador es liberado.
     */
    protected void pointerReleased(int x, int y) {
    }
```

El uso del objeto command "rMonitoreo" requiere la implementación del método commandAction() el cual ejecuta el método cSalTendCanvas() del objeto de clase MidletTitu y que cierra el app.

Código en referencia:

```
    public void commandAction(Command command, Displayable displayable) {
        if (command==rMonitoreo){
            mTend.cSalTendCanvas();
        }
    }
```

Debido a que el punto (0,0) de la pantalla corresponde a la esquina superior izquierda de la misma, en dirección positiva del eje y hacia abajo y eje x hacia la derecha es que requiere la conversión de los puntos. El cruce de las líneas dibujas de los ejes x e y correspondería al punto (0,0). Para ello los métodos conx() y cony() convierten los puntos propios del gráfico a visualizar a los puntos requeridos como datos al dibujar en la pantalla.

Código en referencia:

```
public int conx(int x2){
    x= x2+xof;
    return x;
    }
public int cony(int y2){
    y = gh-1-y2-yof;
    return y;
    }
```

El método estInicial() permite setear el punto inicial a dibujar de la primera línea.

Código en referencia:

```
private void estInicial(){
    x0=1;
    y0=1;
    x1=gwSeg-1;
}
}
```

### 4.3.2 Pantallas del app

Dependiendo de las características de la pantalla del teléfono móvil, las imágenes se muestran de diferentes maneras. Para el caso del teléfono móvil seleccionado las pantallas se ven en el Anexo (B).

## 4.4 Pruebas

### 4.4.1 Pruebas de descubrimiento y emparejamiento

Una vez el teléfono móvil descubre al dispositivo reemplazo se realiza el emparejamiento (pairing). Esto se realiza una sola vez.

### 4.4.2 Pruebas de alcance

Se recurre a un equipo adicional para la medición de la señal del Bluetooth recibida, (Received Signal Strength Indication - RSSI).

Equipo de medición

Tipo: Tablet

Marca: Samsung

Modelo: GT-P7510

Características principales:

- Tecnología Bluetooth.
- Apps instalados: Bluetooth Finder v.1.2 y Bluetooth Signal v.1.0

Las mediciones se han realizado en tres puntos: cercano al equipo, a 1m y 4m de distancia. Todos los resultados son con visión directa (sin obstáculos). No se realizaron pruebas con obstáculos por la diversa variedad de los mismos y por lo cual no es una buena referencia. Los resultados de las mediciones se muestran en el Anexo (C).

## 4.5 Estimación de Costos

### 4.5.1 Hardware y Software

En la Tabla N° D.1 del Anexo (D) se muestra los precios referenciales de los productos a adquirir para implementar el presente trabajo. No se considera el teléfono móvil por ser de uso personal o suministrado por la empresa al personal de comisionamiento. En el caso de software, estos tienen costo cero.

### 4.5.2 Recursos Humanos

En la Tabla N° D.2 del Anexo (D) se muestran las actividades a realizar por cada especialista y a la vez el monto total a pagar, no se considera un salario mensual ya que las actividades son específicas y de plazos cortos. Los montos por Horas-hombre (HH) por personal son referenciales.

### 4.5.3 Comparación de costos entre tecnologías de comunicación inalámbrica

En la Tabla N° D.3 del Anexo (D) se muestra la comparación de costos entre otras tecnologías de comunicación inalámbrica y el Bluetooth. Esto confirma la selección de la tecnología Bluetooth como la mejor alternativa para el presente informe.

## 4.6 Cronograma

Ver Anexo (E)

# CONCLUSIONES Y RECOMENDACIONES

## CONCLUSIONES

1. Los resultados obtenidos dan una buena visión de la implementación del trabajo planteado para aplicarse en el campo industrial, principalmente en la etapa de comisionamiento. Involucra un costo mínimo de inversión para la compañía, así como requerimiento de recursos humanos y el tiempo que acarrea su implementación.
2. El trabajo planteado permite aprovechar las bondades de la tecnología Bluetooth y obtener una mayor productividad en la compañía sin involucrar mayores actividades al personal.
3. El uso de un equipo familiar para el empleado como es el teléfono móvil, así también la facilidad en el uso de la aplicación desarrollada (app) evita rechazos por parte del personal al cambio en la manera de desarrollar sus actividades.
4. Actualmente en los proyectos industriales, el personal utiliza teléfonos móviles inteligentes por lo que la compatibilidad para el uso de la tecnología Bluetooth está garantizado, así también de la aplicación desarrollada (app).
5. Respecto al alcance de la señal de RF Bluetooth, ésta permite dar mayor movilidad al usuario en el momento de las pruebas permitiendo supervisar más puntos críticos del equipo principal. Así también el menor alcance que tiene esta tecnología comparada con el WiFi aumenta la seguridad en el área de uso.
6. El desarrollo de la aplicación bajo una plataforma Java, el uso de una máquina virtual y un convertidor de fuente permite que el mismo código fuente pueda ser usado en la mayoría de teléfonos móviles, así la ejecución de la aplicación desarrollada está garantizada.

## RECOMENDACIONES

1. El dispositivo en serie indicado como requerido presenta una Clase 1 en el alcance de la tecnología Bluetooth, esta características no es necesaria, con una de Clase 2 es suficiente por tanto una mejora del presente trabajo es ver en el mercado si está disponible un dispositivo con esa característica.
2. Se recomienda diseñar el circuito para que el consumo de energía de las otras funciones ofrecidas por el producto y que no son utilizadas sean mínimas. Este punto

sumado al anterior permite el uso de una fuente de alimentación de menor capacidad como podría ser unas pilas tipo botón reduciendo aún más las dimensiones del dispositivo en serie.

# ANEXO A
# CÓDIGOFUENTE DEL APP

BLUETOOTHMAIN

```
/*
 * BLUETOOTHMAIN
 * Clase base, subclase de MIDlet
 */
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;
import java.lang.*;
import java.io.*;
import javax.microedition.io.*;
import javax.bluetooth.*;

/**
 * @author hve
 */
public class BluetoothMain extends MIDlet implements CommandListener, Runnable {


public void startApp() {
   }

public void pauseApp() {
   }

public void destroyApp(boolean unconditional) {
   }
public void commandAction(Command c, Displayable d){
   }
public void run(){

   }
}
```

MIDLETTITU

```
/*
 * MIDLETTITU
 * Subclase de BluetoothMain
 */

import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;
import java.lang.*;
import java.io.*;
import javax.microedition.io.*;
import javax.bluetooth.*;
import java.util.*;

/**
 * @author hve
 */
public class MidletTitu extends BluetoothMain implements CommandListener{
```

```
// Declaracion
private Display display;
private List COMISIONAMIENTO;
private Form SELECCIONANDO;
private List MONITOREO;
private Form VALOR;
private Alert ACTIVARBLUETOOTH;
private Command cSalir;
private ChoiceGroup listDisp;
private TextField tIngrServ;
private Command cConectar;
private int iCanal;
public String sPara,sParaUnit;
private String coneString;
private Command cRegresar;
private StreamConnection coneConector;
public DataInputStream disDatos;
private byte[] bValor;
public int iValor;
public int iTendencia;
private TendCanvas TENDENCIA;
private Timer Secuenciador;
private TareaSeg tareaseg;
private Thread thValor;
private Thread thTendencia;

public MidletTitu(){
try{

display=Display.getDisplay(this);
COMISIONAMIENTO = new List("COMISIONAMIENTO",List.IMPLICIT);
COMISIONAMIENTO.insert(0,"SELECCIONANDO", null);
cSalir=new Command("SALIR",Command.EXIT,1);
COMISIONAMIENTO.addCommand(cSalir);
COMISIONAMIENTO.setCommandListener(this);
    ACTIVARBLUETOOTH=new Alert("ACTIVAR BLUETOOTH","Presionar boton de
reconocimiento \"Done\"",null,AlertType.WARNING);
ACTIVARBLUETOOTH.setTimeout(Alert.FOREVER);

//    Form: SELECCIONANDO

listDisp=new ChoiceGroup("TRANSMISORES",Choice.EXCLUSIVE);
listDisp.append("PT100001", null);
listDisp.append("VT100001", null);
listDisp.append("TT100001", null);
listDisp.append("PT100002", null);
listDisp.append("VT100002", null);
listDisp.append("TT100002", null);
listDisp.setSelectedFlags(new boolean[]{false, false, false, false, false, false});
tIngrServ=new TextField("INGRESAR CANAL DEL SERVIDOR:
","",2,TextField.NUMERIC);
cConectar=new Command("CONECTAR",Command.OK,1);
    SELECCIONANDO=new Form("SELECCIONANDO", new Item[]{listDisp,tIngrServ});
SELECCIONANDO.addCommand(cSalir);
```

```
SELECCIONANDO.addCommand(cConectar);
SELECCIONANDO.setCommandListener(this);

  //    List: MONITOREO
     MONITOREO=new List("MONITOREO",List.IMPLICIT);
MONITOREO.insert(0,"TENDENCIA", null);
MONITOREO.insert(0,"VALOR", null);
MONITOREO.addCommand(cSalir);
MONITOREO.setCommandListener(this);

//        Form: VALOR
    VALOR=new Form("VALOR");
VALOR.addCommand(cSalir);
VALOR.setCommandListener(this);

//      TendCanvas: TENDENCIA
     TENDENCIA=new TendCanvas(this);

//     Timer: Secuenciador
Secuenciador=new Timer();
tareaseg=new TareaSeg(TENDENCIA);
    }
catch (Exception e){
System.out.println("EXCEPCION:      " + e);
    }
    }

public void startApp() {
display.setCurrent(COMISIONAMIENTO);
   }

public void pauseApp() {
   }

public void destroyApp(boolean unconditional) {
try{
disDatos.close();
coneConector.close();
   }
catch(Exception e){
System.out.println("Exception es:"+e);
   }
   }
public void commandAction(Command c,Displayable d){
if (d==COMISIONAMIENTO){
if (c==cSalir){
destroyApp(true);
notifyDestroyed();
          }
if (c==COMISIONAMIENTO.SELECT_COMMAND){
display.setCurrent(ACTIVARBLUETOOTH, SELECCIONANDO);
          }
       }
if (d==SELECCIONANDO){
```

```
if (c==cConectar){
iCanal=Integer.parseInt(tIngrServ.getString());
switch(listDisp.getSelectedIndex()){
case 0:{
sPara = listDisp.getString(0);
sParaUnit="psi";
coneString="btspp://184617e3db11"+":"+iCanal;
//                      PT100001=TABLET
break;}
case 1:{
sPara = listDisp.getString(1);
sParaUnit="mm/s";
coneString="btspp://f40b93b77c22"+":"+iCanal;
//                      VT100001=9700
break;}
case 2: {
sPara = listDisp.getString(2);
sParaUnit = "ºC";
coneString = "btspp://cc52af019c1e"+":"+iCanal;
   //                   TT100001=LAPTOP
break;}
case 3:{
sPara = listDisp.getString(3);
sParaUnit="psi";
coneString="btspp://0018003603f4"+":"+iCanal;
//                      PT100002=TECLADO
break;}
case 4:{
sPara = listDisp.getString(4);
sParaUnit="mm/s";
coneString="btspp://cc55ad26865b"+":"+iCanal;
//                      VT100002=9800
break;}
case 5: {
sPara = listDisp.getString(5);
sParaUnit = "ºC";
coneString = "btspp://"+":"+iCanal;
//                    TT100002=
break;}
            }

//          CONECTARSE CON DISPOSITIVO EN SERIE
//          CREAR EL DATAINPUTSTREAM
new Thread(this).start();
            }
display.setCurrent(MONITOREO);
if (c==cSalir){
destroyApp(true);
notifyDestroyed();
          }
          }
if (d==MONITOREO){
if (c==List.SELECT_COMMAND){
switch(MONITOREO.getSelectedIndex()){
```

```
case 0:{
                    //      IR A LA PANTALLA DE VALOR
thValor=new Thread(new ClassValor(disDatos));
thValor.start();
break;
                    }
case 1:{
                     //      IR A LA PANTALLA DE TENDENCIA
Secuenciador.schedule(tareaseg, 0, 1000);
thTendencia=new Thread(new ClassTendencia(disDatos));
thTendencia.start();
break;
                    }
                }
            }
if (c==cSalir){
destroyApp(true);
notifyDestroyed();
            }
        }
if (d==VALOR){
if (c==cSalir){
destroyApp(true);
notifyDestroyed();
            }
        }
if (d==TENDENCIA){
if (c==cSalir){
destroyApp(true);
notifyDestroyed();
            }
        }
    }

public void run(){
try{
coneConector=(StreamConnection)Connector.open(coneString);
disDatos = new DataInputStream(coneConector.openInputStream());
}
catch(Exception e){
System.out.println("EXCEPCION ES:"+e);
SELECCIONANDO.append("EXCEPCION ES:"+e);
}
    }

public int metLeerDatos(DataInputStream dis1){
byte[] bmetLD=new byte[4];
try{
int iLong=dis1.read(bmetLD);

int imetValor=0;
for(int j=iLong;j>0;j--){
bmetLD[j-1]-=48;
for(int k=iLong-j;k>0;k--){
```

```
bmetLD[j-1]*=10;
            }
imetValor+=bmetLD[j-1];
         }
return imetValor;
      }
catch(Exception e){
return 77;
      }
   }

class ClassValor implements Runnable {
private DataInputStream disClassValor;
public ClassValor(DataInputStream disV){
          this.disClassValor=disV;
      }
public void run(){
display.setCurrent(VALOR);
while (display.getCurrent()==VALOR) {
iValor=metLeerDatos(disClassValor);
iTendencia=iValor;
if (VALOR.size()>0){
VALOR.delete(0);
}
VALOR.append(sPara+"= "+iValor+sParaUnit);
          }
       }
   }

class ClassTendencia implements Runnable {
private DataInputStream disClassTendencia;
public ClassTendencia(DataInputStream disT){
          this.disClassTendencia=disT;
          }

public void run(){
         TENDENCIA.bBack=true;
display.setCurrent(TENDENCIA);
while (display.getCurrent()==TENDENCIA) {
iTendencia=metLeerDatos(disClassTendencia);
iValor=iTendencia;
}
      }
   }

public void cSalTendCanvas (){
destroyApp(true);
notifyDestroyed();
   }
}
```

---

TAREASEG

```
/*
```

```
 * TAREASEG
 * Subclase de TimerTask
 */
import java.util.*;
/**
 *
 * @author hve
 */
class TareaSeg extends TimerTask {
private TendCanvas tCanvas;
public TareaSeg (TendCanvas ttTend){
      this.tCanvas=ttTend;
   }

public final void run(){
      tCanvas.x0=tCanvas.x1;
      tCanvas.x1+=tCanvas.gwSeg;
      tCanvas.y0=tCanvas.y1;
tCanvas.repaint();
}
}
```

---

TENDCANVAS

```
/*
 * TENDCANVAS
 * Subclase de Canvas
*/
import javax.microedition.lcdui.*;
/**
 * @author hve
 */
public class TendCanvas extends Canvas implements CommandListener {
int gh;
int gw;
int x;
int y;
int xof=7;
int yof=7;
int xvar=2;
int yvar=30;
int ghGraf;
int gwGraf;
int xGraf;
int yGraf;
int iValorTend;
int gwSeg=2;
int xSeg;
int ySeg;
int x1;
int y1;
int x0;
int y0;
int nSeg;
```

```
boolean bBack;
private MidletTitu mTend;
private Command rMonitoreo;

public TendCanvas(MidletTitu mLocal) {
      this.mTend=mLocal;
gh=this.getHeight();
gw=this.getWidth();
ghGraf=gh-yof-yvar;
gwGraf=gw-xof-xvar;
try {
rMonitoreo=new Command("SALIR",Command.EXIT,1);
this.addCommand(rMonitoreo);
setCommandListener(this);
bBack=true;
         }
catch (Exception e) {
e.printStackTrace();
      }
estInicial();
   }
   /**
    * paint
    */
public void paint(Graphics g) {
if (bBack==true){
g.setColor(255,255,255);
g.fillRect(0, 0, gw, gh);
g.setColor(0, 0, 0);
g.drawLine(conx(0),cony(0),conx(gwGraf),cony(0));
g.drawLine(conx(0),cony(0),conx(0),cony(ghGraf));
bBack=false;
      }
g.setClip(0,0,gw,yvar);
g.setColor(255,255,255);
g.fillRect(0, 0, gw, gh);
g.setColor(0, 0, 0);
g.drawString(mTend.sPara + "= "+mTend.iTendencia+mTend.sParaUnit, gw-1, 0,
Graphics.TOP | Graphics.RIGHT);
g.setClip(0,0,gw,gh);
g.drawLine(conx(x0),cony(y0),conx(x1),cony(mTend.iTendencia));
y1=mTend.iTendencia;
   }
   /**
    * Llamado cuando una tecla es presionada.
*/
protected void keyPressed(int keyCode) {
}

   /**
    *
    * Llamado cuando una tecla es liberada
*/
protected void keyReleased(int keyCode) {
```

```
}

    /**
     * Llamado cuando una tecla es repetida
*/
protected void keyRepeated(int keyCode) {
}

    /**
     * Llamado cuando el apuntador es arrastrado.
*/
protected void pointerDragged(int x, int y) {
}

    /**
     * Llamado cuando el apuntador es presionado.
*/
protected void pointerPressed(int x, int y) {
}

    /**
     * Llamado cuando el apuntador es liberado.
*/
protected void pointerReleased(int x, int y) {
    }

public void commandAction(Command command, Displayable displayable) {
if (command==rMonitoreo){
mTend.cSalTendCanvas();
        }
    }
public int conx(int x2){
        x= x2+xof;
return x;
        }
public int cony(int y2){
y = gh-1-y2-yof;
return y;
        }
private void estInicial(){
x0=1;
        y0=1;
x1=gwSeg-1
    }
}
```

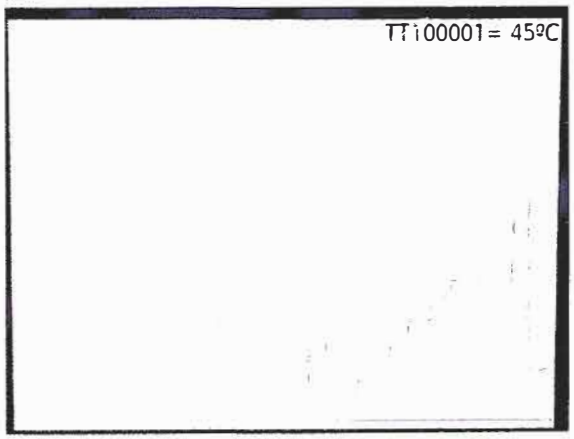
# ANEXO B
# PANTALLAS DEL APP EN EL TELÉFONO MÓVIL

COMISIONAMIENTO
SELECCIONANDO


SELECCIONANDO
TRANSMISORES
123
PT100001
VT100001
• TT100001
PT100002
VT100002
TT100002
INGRESAR CANAL DEL SERVIDOR 1

VALOR
TT100001= 67ºC

MONITOREO
VALOR
TENDENCIA



# ANEXO C
# RESULTADO DE LAS PRUEBAS DE ALCANCE


Bluetooth Signal
BlackBerry 9800
-32
dBm
Enterate de las últimas novedades
Conócenos y se parte de la historia
Pilsen Callao tradicion
7:03 PM


Bluetooth Signal
BlackBerry 9800
-45
dBm
Enterate de las últimas novedades
Conócenos y se parte de la historia
Enterate de las últimas novedades
Conócenos y se parte de la historia
7:04 PM


Bluetooth Signal
BlackBerry 9800
-82
dBm
Enterate de las últimas novedades
Conocenos y se parte de la historia
Enterate de las últimas novedades
Conocenos y se parte de la historia
7:05 PM

# ANEXO D
# ESTIMACIÓN DE COSTOS

**TABLA N° D.1** Estimación de Costos para implementación del trabajo, solo hardware y software [Elaboración propia].

| Productos | Unidad | Cantidad | Precio Unitario (USD $) (*) | Precio Total (USD $) (*) |
|---|---|---|---|---|
| **Hardware** | | | | |
| Dispositivo en Serie | und | 01 | 113.85 | 113.85 |
| Antena del dispositivo | m | 01 | 15.00 | 15.00 |
| Bateria para el dispositivo | und | 01 | 50.00 | 50.00 |
| Circuito acondicionador de la señal de entrada | und | 01 | 30.00 | 30.00 |
| Resina epoxi como alojamiento del dispositivo | 100 gr | 03 | 5.00 | 15.00 |
| **Software** | | | | |
| Netbeans IDE 7.2 | und | 01 | 0.00 | 0.00 |
| Sun Java Wireless Toolkit 2.5.2. | und | 01 | 0.00 | 0.00 |
| | | | TOTAL (USD $) | 223.85 |
| | | | TOTAL (S/.) | 604.40 |

(*) Precios puestos en mercado local.

**TABLA N° D.2** Estimación de Costos respecto a los recursos humanos [Elaboración propia].

| Personal requerido | Actividades | HH (Horas Hombre) | Monto por HH (S/.) | Monto por actividad (S/.) |
|---|---|---|---|---|
| **Técnico Electrónico** | | | | |
| | Preparación del Dispositivo en Serie | 40 | 10.00 | 400.00 |
| | Preparación del acondicionador para señal de entrada | 10 | 10.00 | 100.00 |
| | Ensamble de todas las partes del hardware | 10 | 10.00 | 100.00 |
| | Aplicar la resina epoxi | 05 | 10.00 | 50.00 |
| | | | Monto Total (S/.) | 650.00 |
| **Programador de sistemas** | | | | |
| | Programación del app | 40 | 15.00 | 600.00 |
| | Habilitación de la comunicación | 08 | 15.00 | 120.00 |
| | | | Monto Total (S/.) | 1,370.00 |

**TABLA N° D.3** Estimación de Costos entre otras tecnologías de comunicación inalámbrica y el Bluetooth [Elaboración propia].

| Tecnología de comunicación | Equipos. Contratos e Instalacion | Unidad / Horas-Hombre (HH) | Precio Unitario (USD $) | Precio Total (USD $) |
|---|---|---|---|---|
| **Redes de comunicación para telefonía móvil (GSM, GPRS, HSDPA entre otras)(*)** | | | | |
| | Equipos y contratos | | | |
| | Modem para GSM, GPRS, HSDPA u otra comunicación disponible. | 01 | 400.00 | 400.00 |
| | Contrato de servicio (licencia de uso de la comunicación movil, como referencia de un año) | 01 | 1,200.00 | 1,200.00 |
| | Instalacion | | | |
| | Tecnico electricista (HH) | 10 | 5.00 | 50.00 |
| | Tecnico en telecomunicaciones (HH) | 10 | 5.00 | 50.00 |
| | | | **Monto Total ($)** | **1,700.00** |
| **Wifi/Wlan** | | | | |
| | Equipos y contratos | | | |
| | Access point | 01 | 1,000.00 | 1,000.00 |
| | Soporteria y materiales en general | 01 | 1,000.00 | 1,000.00 |
| | Instalación | | | |
| | Tecnico electricista (HH) | 20 | 5.00 | 100.00 |
| | Tecnico electronico (HH) | 20 | 5.00 | 100.00 |
| | | | **Monto Total ($)** | **2,200.00** |
| **WirelessHart** | | | | |
| | Equipos y contratos | | | |
| | Adaptador WirelessHart | 01 | 1,000.00 | 1,000.00 |
| | Gateway para WirelessHart | 01 | 1,500.00 | 1,500.00 |
| | Access point | 01 | 1,000.00 | 1,000.00 |
| | Soporteria y materiales en general | 01 | 1,000.00 | 1,000.00 |
| | Instalación | | | |
| | Tecnico electricista (HH) | 20 | 5.00 | 100.00 |
| | Tecnico electronico (HH) | 20 | 5.00 | 100.00 |
| | | | **Monto Total ($)** | **4,700.00** |
| **Bluetooth** | | | | |
| | Equipos y contratos | | | |
| | Ver Tabla D.1. | | 223.85 | 223.85 |
| | Instalación | | | |
| | Ver Tabla D.2. | | 507.41 | 507.41 |
| | | | **Monto Total ($)** | **731.26** |

(*) GSM (Global System for Mobile), GPRS(General Packet Radio Service), HSDPA (High Speed Downlink Packet Access).

**ANEXO E**
**CRONOGRAMA**


Diagrama de Gantt
Nombre de tarea
Duración
Comienzo
Fin
1 - 1. Implementacion del trabajo 11 días mié 1/2/13 mié 1/16/13
2 - 1.1. Actividades del tecnico electrónico 10 días mié 1/2/13 mar 1/15/13
3 1.1.1. Preparación del Dispositivo en Serie 5 días mié 1/2/13 mar 1/8/13
4 1.1.2. Preparación del acondicionador para señal de entrada 2 días mié 1/9/13 jue 1/10/13
5 1.1.3. Ensamble de todas las partes del hardware 2 días vie 1/11/13 lun 1/14/13
6 1.1.4 Aplicar la resina epoxi 1 día mar 1/15/13 mar 1/15/13
7 - 1.2. Actividades del programador de sistemas 11 días mié 1/2/13 mié 1/16/13
8 1.2.1. Programación del app 5 días mié 1/2/13 mar 1/8/13
9 1.2.2. Habilitación de la comunicación 1 día mié 1/16/13 mié 1/16/13
sáb dic 29
lun dic 31
mié ene 2
vie ene 4
dom ene 6
mar ene 8
jue ene 10
sáb ene 12
lun ene 14
mié ene 16

# ANEXO F
# HOJA DE DATOS DEL DISPOSITIVO EN SERIE REQUERIDO

ROVING NETWORKS
WIRELESS FOR LESS

# RN-24 / RN25

www.rovingnetworks.com

rn-24-ds v2.3 8/23/2010

## Class 1 Bluetooth® Super Module

### Features

- Bluetooth 2.1/2.0/1.2/1.1 and v2.0+EDR
- Embedded SPP/DUN stack support, no host or processor stacks required
- Small-form factor 24 Pin DIP package (0.1" pitch by 0.9" socket width)
- Accepts 4 Vdc to 24Vdc power (RN-24)
- Hardware TTL, RS- 232 and RS-485 levels
- RS-485 signaling with auto-direction control (RN25 only)
- UART baud rate: 1200bps up to 921.6Kbps
- 9 General Purpose Input/Output Pins (4ma source/sink) controlled via remote commands
- 2 Channel 8 Bit AD converter (5Hz, 0-1.8VDC )
- LEDs indicate connection and RX/TX status
- On board antenna or external SMA jack
- Low power consumption *(30mA connected, <10mA sniff mode)*
- Programmable over local UART and Bluetooth
- Simple ACSII command interface
- UART (SPP or HCI) and USB (HCI only) data connection interfaces
- Sustained SPP data rates - 240Kbps (slave), 300Kbps (master)
- Embedded Bluetooth stack profiles included *(requires no host stack)*: GAP, SDP, RFCOMM and L2CAP protocols, with SPP and DUN profile support.

### Applications

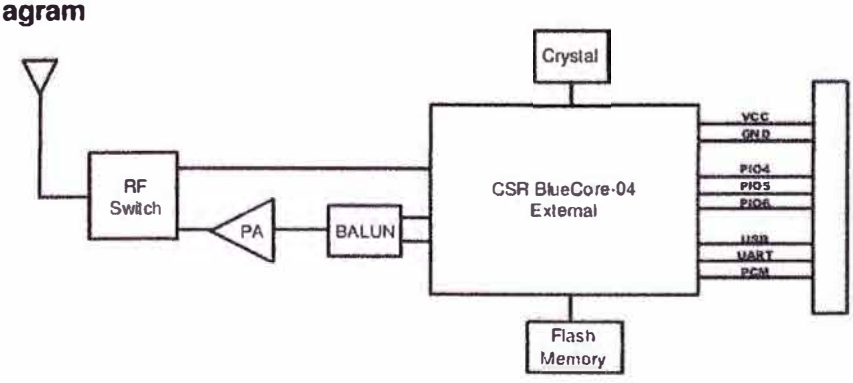
- Wireless RS232/RS485 cable replacement
- Remote equipment monitoring
- Scanners, GPS and measurement systems
- Industrial sensors and controls
- Medical devices

### Description

The RN24 transforms your product's serial interface into a wireless connection. The RN24 is a Class 1 Bluetooth radio with auto-connect and connect on RX data features, enabling your product to transfer data up to 100M. The RN24 accepts a wide range of unregulated DC power. The DIP packaging with GPIO, UART interfaces and AD converter makes it the perfect prototyping module and allows products to have optional Bluetooth capability when the module is plugged in.

The RN-25 has the same powerful Bluetooth radio and simple ASCI command interface as the RN-24 but has a RS-485 hardware interface rather than RS-232.

### Block Diagram



ROVING NETWORKS
WIRELESS FOR LESS



## Overview

- UART baud rate speeds: 1200bps up to 921Kbps, non-standard baud rates can be programmed.
- Class 1 radio, 330' (100m) distance, 12dBm output transmitter, -80dBm typical receive sensitivity
- Frequency 2402 ~ 2480MHz,
- FHSS/GFSK modulation, 79 channels at 1MHz intervals
- Secure communications, 128 bit encryption
- Error correction for guaranteed packet delivery
- UART local and over-the-air RF configuration
- Auto-discovery/pairing requires no software configuration (instant cable replacement).
- Auto-connect master, IO pin (DTR) and character based trigger modes

## Environmental Conditions

| Parameter | Value |
|---|---|
| Temperature Range (Operating) | -40°C ~ 85°C |
| Temperature Range (Storage) | -40°C ~ 85°C |
| Relative Humidity (Operating) | ≤90% |
| Relative Humidity (Storage) | ≤90% |

## Electrical Characteristics

| **Power supply voltage RN24** *WARNING – Only one source of power must be supplied.* | | 3.3VDC nominal or 5-24VDC unregulated | | |
|---|---|---|---|---|
| **Parameter** | **Min** | **Typ.** | **Max.** | **Unit** |
| Supply Voltage (DC) using pin 3 | 3.0 | 3.3 | 3.6 | V |
| Supply Voltage (DC) using pin 12 | 5 | - | 24 | V |
| RX Supply Current | | 35 | 60 | mA |
| TX Supply Current | | 65 | 100 | mA |

**NOTE: do not exceed 5 VDC on the RN-25 or permanent damage to the module will occur!**

| **Power supply voltage RN25** | | 5 VDC unregulated on pin12 | | |
|---|---|---|---|---|
| **Parameter** | **Min** | **Typ.** | **Max.** | **Unit** |
| Supply Voltage (DC) using pin 3 | - | - | - | |
| Supply Voltage (DC) using pin 12 | 5 | - | 5 | V |
| RX Supply Current | | 35 | 60 | mA |
| TX Supply Current | | 65 | 100 | mA |

| **Average power consumption** | **Min** | **Typ.** | **Max.** | **Unit** |
|---|---|---|---|---|
| Standby/Idle (default settings) | | 25 | | mA |
| Connected (normal mode) | | 30 | 50 | mA |
| Connected (low power Sniff) | | 8 | | mA |
| Standby/Idle (lowest power) | 250uA | 2.5 | | mA |

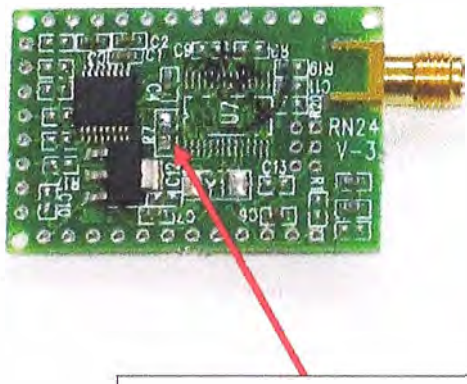
## Digital I/O Characteristics

*WARNING: Take care not to exceed the voltage limits to the VCC, TTL SERIAL, and GPIO pins. 0 negative voltage or voltage exceeding 3.30 VDC can permanently damage the device!*

- Use a 10KΩ series resistor on inputs
- GPIO sink current is 4mA max.
- Unused pins should float

| 2.7V ≤ VDD ≤ 3.0V | Min | Typ. | Max. | Unit |
|---|---|---|---|---|
| Input logic level LOW | -0.4 | - | +0.8 | V |
| Input logic level HIGH | 0.7VDD | - | VDD+0.4 | V |
| Output logc level LOW | - | - | 0.2 | V |
| Output logic level HIGH | VDD-0.2 | - | - | V |
| All I/O's (except reset) default to weakpull down | +0.2 | +1.0 | +5.0 | uA |

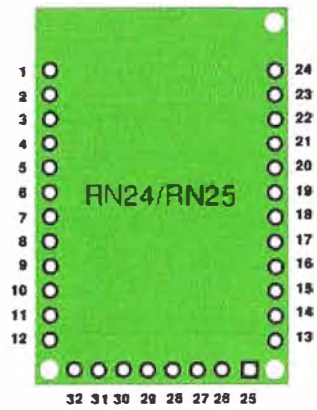
The RN24 can drive either UART signals on the TTL pins 8-11 or RS232 pins 13-16. WARNING: Driving the TTL signals at RS232 levels can permanently damage the module. By default the module is configured to drive only the RS232 pins. To drive the TTL signals you must remove the resistor R7. R7 can be found on the bottom of the board, near the center.




Remove R7 to enable the UART TTL signaling on pins 8-11. This disconnects pins 13-16 from the UART interface.

## Pin Description

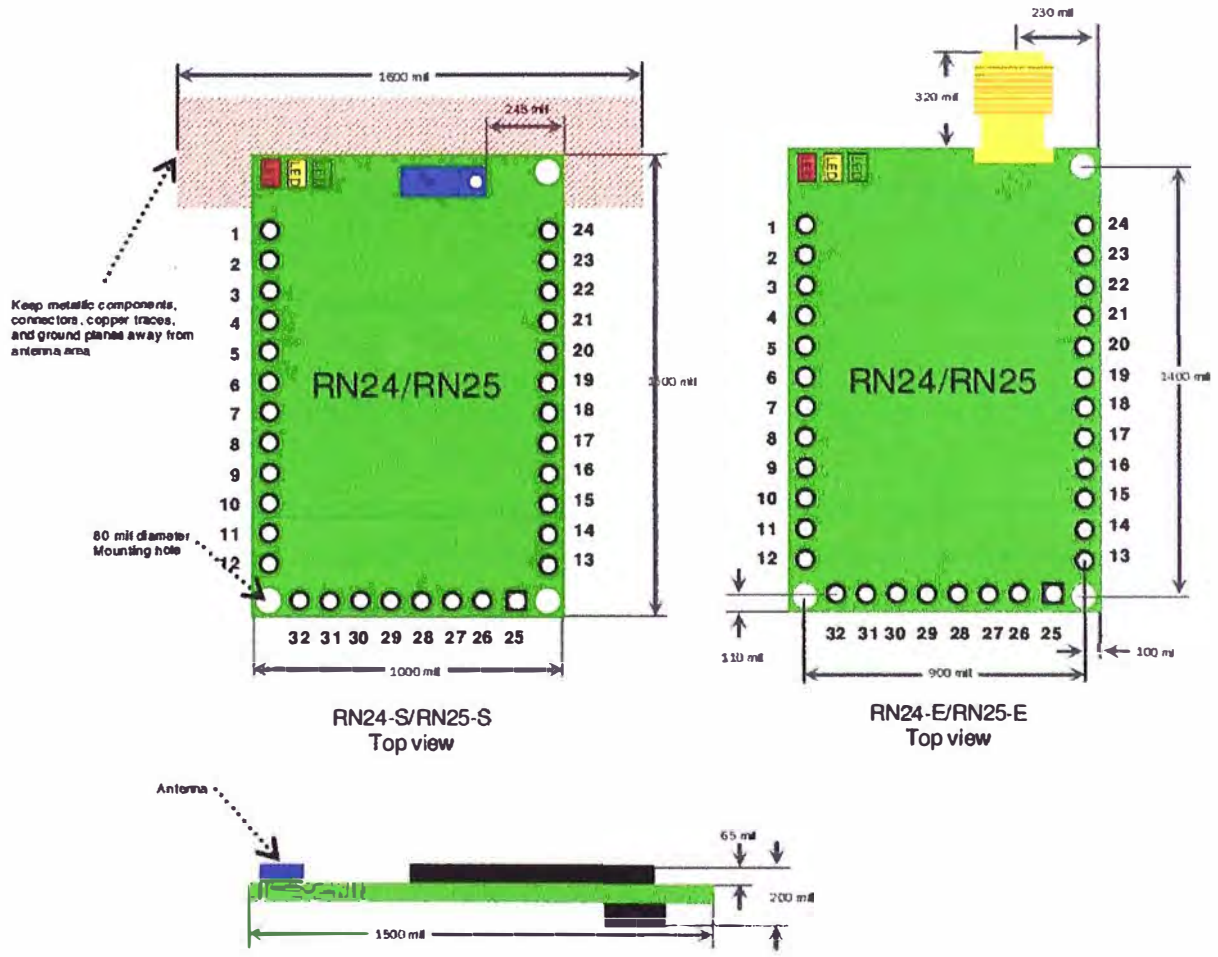
NOTE: The Power can be applied to either pin 12, or pin 3, but **NOT** both. If pin 12 is powered then Pin 3 can be used as a 3.3V regulated supply output up to 200mA.



Top view

| Pin | Name | Description | Default |
|---|---|---|---|
| 1 | RESET | Active HIGH | Has 1K pulldown |
| 2 | GND | | |
| 3 | VCC | 3.3VDC power | Mutually exclusive with Pin 12 |
| 4 | SPI_CS | Programming Only | No Connect |
| 5 | SPI_CK | Programming Only | No Connect |
| 6 | SPI_MO | Programming Only | No Connect |
| 7 | SPI_MI | Programming Only | No Connect |
| 8 | CTS | Input, if set HIGH disables transmitter | TTL input |
| 9 | TX | Transmit, output from the module | TTL output |
| 10 | RTS | Output, goes HIGH to disable host transmitter | TTL output |
| 11 | RX | Receive, input to the module | TTL input |
| 12 | VDD | RN-24 : 5-24V unregulated DC power<br>RN-25 : 5 V unregulated DC power | Mutually exclusive with Pin 3 |
| 13 | UART_TX | UART Transmit, output from the module | RS-232 level signaling |
| 14 | UART_RX | UART Receive, input to the module | RS-232 level signaling |
| 15 | UART_RTS | Output, goes HIGH to disable host transmitter | RS-232 level signaling |
| 16 | UART_CTS | Input, if set HIGH disables module transmitter | RS-232 level signaling |
| 17 | GPIO-7 | Default baud rate, HIGH = 9600, LOW = 115K | Input to RN24, default = LOW |
| 18 | GPIO-6 | Enable auto master mode = HIGH | Input to RN24, default = LOW |
| 19 | GPIO-4 | Restore factory defaults | Set HIGH then toggle for 5 msec. 3 times |
| 20 | GPIO-3 | Enable instant cable mode = HIGH | Input to module, default = LOW |
| 21 | GPIO-2 | Connection status, connected=HIGH | Output from module |
| 22 | RS-485 D+ | RS485 signaling **(Only on RN25)** | (Auto direction detect) |
| 23 | RS-485 D- | RS485 signaling **(Only on RN25)** | (Auto direction detect) |
| 24 | GND | Ground | |
| 25 | GPIO-11 | | |
| 26 | GPIO-8 | Controls red LED | |
| 27 | GPIO-9 | Controls yellow LED | |
| 28 | GPIO-10 | | |
| 29 | AD1 | Optional analog input 0-1.8V | Requires special firmware |
| 30 | AD2 | Optional analog input 0-1.8V | Requires special firmware |
| 31 | NC | | |
| 32 | GND | Ground | |

## Module Dimensions




Connect with 24 pin DIP (through-hole) 0.1" pitch and 0.9" wide socket
Single row mating sockets use:
Digikey P/N: ED7012-ND (female)
ED7512-ND (male)

## Design Concerns

1. ***Reset circuit***. RN-24 contains a 1k pulldown, the polarity of reset is ACTIVE HIGH. A power on reset circuit with delay is OPTIONAL on the reset pin of the module. It should only be required if the input power supply has a very slow ramp, or tends to bounce or have instability on power up. Often a microcontroller or embedded CPU IO is available to generate reset once power is stable

2. ***Factory reset PIO4.*** It is a good idea to connect this pin to a switch, or jumper, or resistor, so it can be accessed. This pin can be used to reset the module to FACTORY DEFAULTS and is often critical in situations where the module has been mis-configured.

3. ***Connection status.*** PIO2 is an output which directly reflects the connection state, it goes HIGH when connected, and LOW otherwise.

4. ***RS485 support.*** Must supply 5V on pin 12 if using the RS485 interface. ***Do not exceed 5VDC or permanent damage to the module will occur!***

   RS-485 signaling with auto-direction control is ONLY supported on PCB rev 4 and later. Treat pins 22 and 23 as no connect on previous PCB boards.

5. ***Using SPI bus for flash upgrade.*** While not required, this bus is very useful for configuring advanced parameters of the Bluetooth modules, and is required for upgrading the firmware on modules. The suggested ref-design shows a 6pin header which can be implemented to gain access to this bus. A minimum-mode version could just use the SPI signals (4pins) and pickup ground and VCC from elsewhere on the design.

6. ***Minimizing Radio interference.*** When integrating the RN24 DIP module with on board chip antenna besure the area around the chip antenna end of RN24-S module protrudes at least 5mm from the PCB and any metal enclosure. If this is not possible use the RN24-E.

7. ***Soldering Reflow Profile.***
   - Lead-Free Solder Reflow
   - Temp: 230 degree C , 30-40 seconds, Peak 250 degree C maximum.
   - Preheat temp: 165 +- 15 degree C, 90 to 120 seconds.
   - Time: Single Pass, One Time

8. ***Connecting to the GPIO.*** Placing 3.3Vdc into the PIO's while they are set as outputs will permanently damage the radio. The failure mode is short across GND and VCC. Use a 10KO resistor is series or a 10KO pull up resistor for input and output PIO's respectively.
   - Make sure to connect a common ground when using the external TX, RX inputs on the 0 – 3.3Vdc
   - For a 3 wire DB-9 interface (tx, rx, gnd only) connect/short CTS to RTS, Factory default is hardware flow control enabled CTS and RTS connected.
   - When using a 5.0Vdc Input, PIO's require a 10K ohm series resistor. PIO's are 0-3.3Vdc not 5 volt tolerant.

## Compliance Information

- FCC Certified, FCC ID T9J-RN24, IC number6514A-RN24
- Environmentally friendly RoHS compliant

# ANEXO G
# GLOSARIO DE TÉRMINOS

**API (Application Programming Interface):** Grupo de funciones y procedimiento utilizados para el desarrollo de aplicaciones.

**Backlisting:** Es la generación de una lista aplicado a las bandas de RF para disminuir las interferencia entre señales.

**BYOD (Bring Your Own Device):** Tendencia actual donde el empleador hace uso de su equipo personal en el desarrollo de sus labores.

**Canvas:** Clase utilizada en la programación para la elaboración de interfaces graficas de bajo nivel.

**CPU (Central Processing Unit):** Componente principal de una computadora como por ejemplo.

**CVM (Compact Virtual Machine):** Software de mayor requerimiento de memoria pero de mayores bondades. Es usado por dispositivos de una mayor funcionalidad que los teléfonos móviles.

**DCS (Distributed Control System):** Equipo de uso industrial para el control de una planta, presenta mayores bondades que el PLC para aplicaciones complejas.

**EDR (Enhanced Data Rate):** Tecnología aplicada a la comunicación Bluetooth para aumentar la velocidad de transmisión de datos.

**GNU:** Proyecto que tiene como objetivo el desarrollo de un sistema operativo libre.

**GNU GPL (Gnu General Public License):** Licencia que permite al usuario usarlo, modificarlo y distribuirlo en forma gratuita.

**HART (Highway Addressable Remote Transducer):** Es estándar en la comunicación con instrumentación inteligente de campo.

**HCF (HART Communication Foundation):** Organización que estandariza las futuras soluciones de comunicación inalámbrica en base al protocolo HART.

**HMI (Human Machine Interface):** Usado para dar referencia a la interacción entre humanos y máquinas, se aplica a sistemas de Automatización de procesos.

**IEC (International Electrotechnical Commission):** Organización encargada de generar estándares en el campo eléctrico, electrónico y afines.

**IP (Internet Protocol):** Protocolo utilizado para transferencia de datos.

**IrDA (Infrared Data Association):** Organización que define la comunicación inalámbrica por rayos infrarrojos.

**J2ME (Java 2 Micro Edition):** Plataforma Java utilizada para el desarrollo de aplicación a instalar en equipos móviles.

**JABWT (Java APIs for Bluetooth Wireless Technology):** Es un grupo de estándares para el desarrollo de API basados en Java y aplicados a la tecnología Bluetooth, también llamado JSR-82.

**JSR (Java Specification Request):** Documentación que describe las especificaciones y tecnologías agregadas a la plataforma *Java*.

**JVM (Java Virtual Machine):** Software utiliza para soportar aplicaciones Java en plataformas distintas.

**KVM (Kilo Virtual Machine):** Software utilizado por aplicaciones generadas en Java para ser usado en equipos móviles de diferente plataforma.

**LE (Low Energy):** Tecnología aplicada a la comunicación Bluetooth para disminuir el consumo requerido por esta.

**Multithread:** Múltiples ejecuciones de procesos de una o varias aplicaciones en determinados periodos de tiempo.

**OBEX (OBject Exchange):** Protocolo usado en la tecnología Bluetooth para la transmisión de datos.

**OSI (Open System Interconnection):** Propuesta realizada por la Organización Internacional para la Estandarización (ISO) con fines de estandarización de interconexión de sistemas abiertos.

**PCMCIA (Personal Computer Memory Card International Association):** Asociación de compañias desarrolladoras de tarjetas de memoria para computadoras.

**PLC (Programmable Logic Controller):** Equipo utilizando en la industria para el control de una planta.

**PPP (Peer to Peer Protocol):** Permite una comunicación entre dos dispositivos con el intercambio de funciones entre cliente y servidor.

**RFCOMM (Radio Frequency Communication):** Protocolo utilizado en la tecnología Bluetooth como reemplazo de la comunicación serial RS-232.

**RS-232:** Interface desarrollada para la transferencia de datos a través de un cable. La comunicación se realiza en forma serial.

**SIG (Special Interest Group):** Organización encargada del desarrollo de la tecnología Bluetooth, implementar y comercializarla entre los miembros del grupo.

**TCP (Transmission Control Protocol):** Protocolo utilizado en el paso de información en la Internet que requiere confirmación de recepción.

**USB (Universal Serial Bus):** Estándar utilizando en la mayoría de equipos electrónicos para la transferencia de datos.

**UART (Universal Asynchronous Receiver-Transmitter):** Dispositivo para la transferencia de datos en ambos sentidos, trabaja como transmisor y receptor.

**UDP (User Datagram Protocol):** Utilizado como alternativa al TCP. No requiere confirmación de recepción de datos.

**WiTECK (Wireless Industrial Technology Konsortium):** Consorcio entre varias marcas reconocidas en el mercado industrial con fines de desarrollar una comunicación inalámbrica.

**WAE (Wireless Application Environment):** Entorno para el desarrollo de aplicaciones a instalar en dispositivos móviles.

**WAP (Wireless Access Protocol):** Impulsado por el sector de telecomunicaciones para prestar servicios a los dispositivos móviles.

**WWW (World Wide Web):** Red a nivel mundial utilizada para la distribución de información.

**WLAN (Wireless Local Area Network):** Sistema de comunicación inalámbrico de tipo flexible utilizado como alternativa a las redes cableadas de área local.

**WIFI (Wireless Fidelity):** Llamado también WLAN.

## BIBLIOGRAFÍA


[1] "Mobile Industrial Worker", Peter Granger, Control Engineering, Marzo 2012.
[2] "802.11 Wireless LAN Fundamentals", Pejman Roshan, Jonathan Leary, Cisco Press, 2003.
[3] "Wi-Fi and Bluetooth coexistence", Sue White, http://www.ecnmag.com/articles/2012/03/wi-fi-and-bluetooth-coexistence, 2012.
[4] "IEEE 802.11:Wireless LANs from a to ñ", William Stallings, IT Pro, Octubre 2012.
[5] "WirelessHART: Applying Wireless Technology in Real-Time Industrial Process", Control", Jianping Song, Song Han, Aloysius K. Mok, Deji Chen, Mike Lucas, Mark Nixon y Wally Pratt, IEEE Computer Society, 2008.
[6] "WirelessHART Smart Wireless Solutions", Aaro Lehto, Emerson Process Management Oy, 2009.
[7] "Emerson Wireless Security – WirelessHart and Wi-Fi Security", Emerson Process Management, 2011.
[8] "Bluetooth Application Programming with the Java API's", Thompson, Kline y Kumar, Ed. Elsevier, 2008.
[9] "High pulse drain impact on CR2032 coin cell battery capacity", Kjartan Furset y Peter Hoffman, Nordic Semiconductor y Energizer, 2011.
[10] "Bluetooth for Programmers", Albert Huang y Larry Rudolph, 2005.
[11] "Bluetooth Specifications and Profiles", Bluetooth SIG, http://www.bluetooth.org/Building/HowTechnologyWorks/ProfilesAndProtocols/Overview.htm, 2012.
[12] "Java APIs for Bluetooth Wireless Technology (JSR-82)", Motorola, 2008.
[13] "Core Architecture Blocks", Bluetooth SIG, http://www.bluetooth.org/Building/HowTechnologyWorks/Architecture/Overview.htm.
[14] "Java a tope: J2ME (Java 2 Micro Edition)", – Sergio Galvez Rojas, Lucas Ortega Diaz)
[15] "Java: The Complete Reference", Herbert Schildt, McGraw-Hill, 2007.
[16] "Java Timeline", Oracle, http://oracle.com.edgesuite.net/timeline/java/, 2011.
[17] "Android Activations - 2012", http://www.android.com/, 2012.
[18] "Introduction to Android Programming", Randeep Bhatia.
[19] "App Development for Mobile Devices", Jaerock Kwon, Kettering University, 2011.
[20] "Introducción al desarrollo de aplicaciones para Android", Jose Luis Bugarin Peche, Iluminatic SAC.
[21] "Computo Móvil: Introducción a Android", Gregorio Toscano Pulido, Centro de Investigación y de Estudios Avanzados del IPN.
[22] "The Google Android Stack", Dominik Gruntz y Carlo Nicola, IMVS, 2007.
[23] "Introducción a Android", Manuel Báez, Álvaro Borrego, Jorge Cordero, Luis Cruz, Miguel González, Francisco Hernández, David Palomero, José Rodríguez de Llera, Daniel Sanz, Mariam Saucedo y Pilar Torralbo, Álvaro Zapata.
[24] "Timeline of Android versions", Wikipedia, http://en.wikipedia.org/wiki/Template:Timeline_of_Android_versions, 2012.
[25] "Open Handset Alliance", http://www.openhandsetalliance.com/,
[26] "An Introduction to Android", Huang Xuguang, Database Lab. Inha University, 2009.
[27] "iOS Technology Overview", Apple Inc., 2012.

[28] "Introduction to iOS Development", Dave Verwer, 2011.
[29] "iOS version history", Wikipedia, http://en.wikipedia.org/wiki/IOS_version_history, 2012.
[30] "Introduction to iOS", Kenneth M. Anderson, 2011.
[31] "The iOS 4 Architecture and SDK Frameworks", Techotopia, http://www.techotopia.com/index.php/The_iOS_4_Architecture_and_SDK_Frameworks, 2011.
[32] "Apple agrees to license for Swiss railway clock in iOS 6", Jon Fingas, Engaget, http://www.engadget.com/2012/10/12/apple-licenses-swiss-railway-clock-knows-what-time-it-is/, 2012.
[33] "What is BlackBerry OS?", Miles J Thomas, http://www.3g.co.uk/PR/June2012/what-is-blackberry-os.html, 2012.
[34] "Rim History", Blackberry, http://www.blackberry.com/select/get_the_facts/pdfs/rim/rim_history.pdf, 2006.
[35] "Blackberry Architecture", Anthony Scian, Blackberry.
[36] "RIM 850 4MB", GSM2Indonesia, http://gsm2indonesia.wordpress.com/2011/07/25/rim-850-4mb/, 2011.
[37] "RIM's roller coaster", CBC, http://www.cbc.ca/news/interactives/timeline-rim/timeline/timeline.html, 2012.
[38] "BlackBerry 10 OS will have multi-layered security model", John Cox, NetworkWold, http://www.networkworld.com/news/2012/050812-blackberry10-security-259080.html, 2012.
[39] "Blackberry Support Community", Blackberry, http://supportforums.blackberry.com/t5/BlackBerry-App-World-Development/Blackberry-processor-target/td-p/258910, 2009.
[40] "BlackBerry Operating System", CMER Centre for Mobile Education and Research, http://www.qnx.com/products/neutrino-rtos/neutrino-rtos.html.
[41] "Project Management and Control", Subhash Chandra Das, PHI Learning Private Limited, 2012.
[42] "Construction Planning and Management", P S Gahlot, New Age International, 2002.
[43] "Changing Your Mind: On the Contributions of Top-Down and Bottom-Up Guidance in Visual Search for Feature Singletons", Jeremy M. Wolfe, Serena J. Butcher, Carol Lee y Megan Hyle, American Psychological Association, 2003.
[44] "Android and iOS to surpass BlackBerry as top enterprise devices for the first time", Lisa Eadicicco, DigitalTrends, 2012.
[45] "Java APIs for Bluetooth Wireless Technology (JSR-82)", Motorola, 2008.