Plataforma domótica basada en la integración de un hipervisor con Android-x86

José Félix, Antonio Gutiérrez, Apolinar González, Walter Mata,
Facultad de Ingeniería Mecánica y Eléctrica Universidad de Colima, Av. Universidad # 333, Colima, México, {jfelix, jose_gutierrez, apogon, wmata}@ucol.mx

***Resumen*—El presente artículo describe el análisis y diseño de una plataforma domótica que consta de la integración del hipervisor XtratuM, con un sistema operativo basado en el kernel de GNU/Linux. El hipervisor nos proporciona múltiples niveles independientes de seguridad (MILS por sus siglas en inglés) que permite la ejecución del sistema junto con otras aplicaciones sin que se afecten entre sí por errores. El sistema operativo basado en Linux seleccionado para la integración con el hipervisor por su interfaz gráfica es Android, terminada la unión del Sistema Operativo Móvil con el hipervisor, al mismo tiempo se desarrolla una interfaz de usuario para el control y la automatización correspondiente al sistema domótico de una manera fácil e intuitiva para el usuario.**

***Palabras Clave*—Hipervisor, Tiempo Real, Android, XtratuM, virtualización, domótica.**

I. Introducción

La presencia cada vez mayor de sistemas embebidos en productos y servicios crea enormes oportunidades en diferentes dominios de aplicación. Los sistemas embebidos desempeñan un papel importante no sólo en los productos electrónicos de consumo, sino también en muchos sistemas de seguridad crítica. Por tal motivo, existe un creciente interés científico tanto en las herramientas conceptuales como prácticas para el desarrollo de sistemas embebidos. Dichos sistemas están creciendo en popularidad al existir una gran variedad de aplicaciones haciendo cada vez nuestras vidas más dependientes de los mismos. Más del 98% de los procesadores que se utilizan actualmente están dentro de sistemas embebidos, los cuales prácticamente son invisibles para los consumidores. Esto incluye no sólo aplicaciones de seguridad crítica tales como controladores de dispositivos para aviones, automóviles, ferrocarriles, industria aeroespacial, salud y dispositivos médicos, sino también en comunicaciones, sistemas móviles, medio ambiente, automatización de casas, teléfonos móviles, PDAs, reproductores de DVD, cámaras, etc. teniendo un impacto amplio en la sociedad, incluida la seguridad, privacidad, y los estilos de vida y de trabajo. [1]–[5]

Otros avances importantes dentro del dominio de los sistemas embebidos se da en la virtualización que consiste en la abstracción de los recursos de la computadora, tales como el Microprocesador, memoria RAM, discos duros, tarjetas de red, etc. La cual al tomar estos recursos puede dividir y simular que existe un nuevo hardware para que sea utilizable por otro Sistema Operativo (SO) o aplicación que necesite dichos recursos, creando la ilusión que se está usando más de una computadora en un mismo equipo. La mayoría de los avances recientes en la virtualización se han hecho en los sistemas de escritorio, Xen [6] [7], VMware [8] y Sun VirtualBox [9] son tres ejemplos que están trabajando en el desarrollo de estas tecnologías.

Dentro de los sistemas embebidos generalmente se encuentran SO los cuales son definidos como una capa de software que permite multiplexar abstracciones de hardware para las aplicaciones como memoria volátil, ciclos de procesador, dispositivos de entrada/salida, etc.,. Un SO embebido debe realizar las operaciones expuestas anteriormente, pero en un ambiente donde las aplicaciones poseen numerosas restricciones, particularmente en cuanto a consideraciones de tiempo y energía [10]. Aquí es donde Android entra en juego siendo un SO que puede adaptarse para funcionar en casi cualquier sistema embebido. Un SO para un sistema embebido es usualmente diseñado para una aplicación específica, y por lo tanto es más estático que un sistema operativo de propósito general como el que usamos en nuestras computadoras.

XtratuM puede considerarse como una pequeña parte de la capa más baja del sistema operativo que controla las interrupciones del procesador, controla al manejador de memoria, controla los procesos y que tiene un manejador de particiones. [6]

Ahora Android es un SO para dispositivos móviles (celulares, PDAs, tablets, etc.) desarrollado principalmente por Google y sigue la filosofía de código abierto, por lo tanto, cualquier persona puede descargar el código fuente, modificarlo dependiendo las necesidades y compartir los cambios con la comunidad. Entre 2008 y 2009 se presentaron las primeras versiones del SO, siendo Abril de 2009 la llegada de Android 1.5 (CupCake) empezándose a popularizar [15]; usa el kernel de Linux como capa de abstracción del hardware, es decir, el kernel se usa para que los componentes de Android, la máquina Dalvik, las librerías y las aplicaciones se ejecuten sobre un entorno estandarizado que no dependa del hardware.

Para desarrollar aplicaciones en tiempo real utilizamos el

S.O. PaRTiKle desarrollado principalmente para plataformas embebidas que obedece la norma de POSIX PSE51. PaRTiKle se puede construir para funcionar en tres entornos de ejecución distinto, como un sistema stand-alone, como un proceso regular de Linux o como una partición de XtratuM. Esta última es la opción que nos ayudara a tener aplicaciones en tiempo real junto con la interfaz gráfica amigable de Android.

Dentro del presente artículo podremos encontrar una pequeña introducción y arquitectura general del hipervisor XtratuM además de un resumen de la arquitectura general del Sistema Operativo Móvil Android, conjuntamente hay una explicación de cómo será la plataforma global para aplicaciones domóticas, el trabajo realizado para que XtratuM pueda soportar Android como SO, también un ejemplo pequeños de como seria la interfaz gráfica de la aplicación para el control de la plataforma domótica, con la explicación de por qué se debe de usar un hipervisor en lugar de usar una aplicación nativa de Android.

II. Arquitectura XtratuM

XtratuM es un hipervisor desarrollado por el Instituto de Automática e Informática Industrial de la Universidad Politécnica de Valencia, que proporciona un marco para ejecutar varios sistemas operativos (tiempo real) en un entorno particionado y robusto que ha sido parcialmente financiado por el proyecto FRESCOR Europea y el Centro Nacional de dEtudes Espaciales de Francia (CNES, la agencia espacial francesa), y que se utilizará en el procesador LEON2.

XtratuM se puede utilizar para construir Multiples Niveles independientes de seguridad (Multiple Independent Levels of Security por sus siglas en inglés MILS) de la arquitectura [13]. La característica más interesante de XtratuM es su capacidad de compartir el mismo hardware entre varios sistemas operativos ejecutándose en forma concurrente [14].

Con el fin de ejecutar varios dominios concurrentemente, cada dominio tiene que ser portado a la infraestructura de XtratuM. En particular, los controladores de interrupción y timers serán reemplazados por las llamadas a la API de XtratuM.

XtratuM ha sido implementado siguiendo el enfoque monolítico, ejecutando todos sus servicios en modo privilegiado de procesador y en un espacio de memoria único, siendo todos los servicios accesibles directamente desde cualquier parte del hipervisor, la Fig. 1 muestra la arquitectura interna de XtratuM.

La idea principal detrás del diseño de la arquitectura de XtratuM, es la de virtualizar lo menos posible las partes del hardware para llevar a cabo la ejecución concurrente de varios sistemas operativos; Sin embargo, a diferencia de otros nanokernels (por ejemplo los kernels de la familia L4). Cada sistema operativo debe ser consciente de cómo utilizar las partes del microprocesador y memoria que no han sido virtualizadas.

III. Arquitectura Android

Android es un sistema derivado del kernel de GNU/Linux y fue diseñado especialmente para ambientes de dispositivos móviles y con capacidades de touch screen aunque esto no lo limita a funcionar en otro tipo de arquitecturas o plataformas con otros recursos como x86, arquitectura que será utilizada en esta implementación.

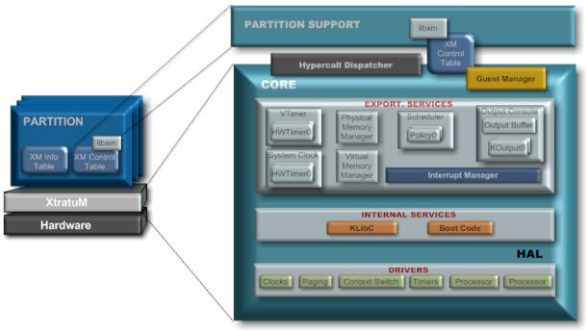

Fig. 1: Arquitectura interna de XtratuM

Como se muestra en la Fig. 2, los componentes que constituyen Android se apilan en capas. Cada una de estas capas manipula elementos de la capa inferior para ejecutar sus funciones. Por ese motivo, a este tipo de arquitectura se le llama pila. Esta es la pila software de Android:

A. *Kernel de Linux*

El núcleo del sistema operativo Android es un kernel Linux versión 2.6, similar al que se incluye en cualquier distribución de Linux, sólo que acondicionado a las características del hardware en el que se correrá Android.

Proporciona una capa de abstracción para el hardware a los que tienen que acceder las aplicaciones. Esto permite que se pueda acceder a esos componentes sin necesidad de conocer el modelo o características precisas de los que están instalados en el teléfono.

B. *Librerias*

La capa que se sitúa sobre el kernel la componen las librerías nativas de Android. Estas librerías están escritas en C o C++ y compiladas para la arquitectura hardware específica del dispositivo. Su cometido es proporcionar funcionalidad a las aplicaciones, para tareas que se repiten con frecuencia, evitando tener que codificarlas cada vez y garantizando que se llevan a cabo de la forma más eficiente.

C. *Entorno de ejecución*

El componente principal del entorno de ejecución de Android es la máquina virtual Dalvik, que ejecuta todas las aplicaciones no nativas de Android, que se compilan en un formato específico para la máquina virtual Dalvik, que es la que las ejecuta. Esto permite compilar una única vez las aplicaciones y distribuirlas teniendo la total seguridad de que podrán ejecutarse en cualquier dispositivo Android con la misma versión del sistema operativo.

D. *Marco de aplicación*

La siguiente capa la establecen todas las clases y servicios que utilizan directamente las aplicaciones, para realizar sus funciones y que, obviamente, se apoyan en las librerías y en el entorno de ejecución. La mayoría de los componentes de esta capa son librerías Java que acceden a los recursos a través de la máquina virtual Dalvik.

E. Aplicaciones

La capa superior la forman las aplicaciones. En esta capa se incluyen todas las aplicaciones del dispositivo, tanto las que tienen interfaz de usuario como las que no, tanto las nativas como las administradas, tanto las que vienen de serie con el dispositivo como las instaladas por el usuario [11].

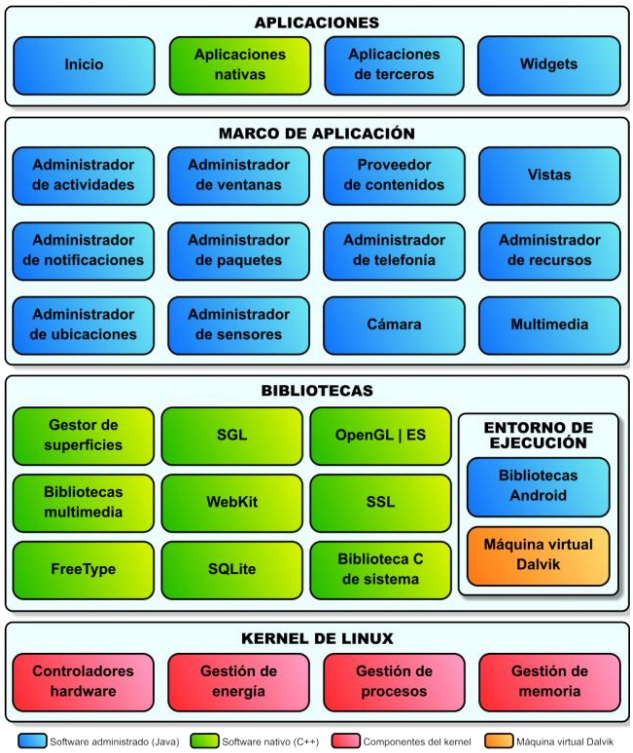

Fig. 2: Arquitectura de Android

IV. Plataforma Global para las Aplicaciones Domóticas

En la Fig. 3 se puede apreciar la arquitectura global de la plataforma domótica donde el hipervisor se ejecuta directamente en la arquitectura de hardware. El hipervisor completamente aislado de los distintos ámbitos, permite la comunicación a través de las colas de mensajes, hiperllamadas y memoria compartida. Como resultado, los distintos sistemas operativos se pueden ejecutar de forma independiente y el hipervisor en tiempo real, controla la ejecución de cada uno de ellos. El Middleware basado en el patrón P/S (por sus siglas en inglés Publish/Subscribe) del DDS implementa un subconjunto de las interfaces estándar de esta especificación, para realizar publicaciones y subscripciones en las aplicaciones domóticas desarrolladas. Estas aplicaciones pueden difundir y recolectar información a través de una interfaz P/S proporcionada por el middleware. La plataforma global muestra particiones diferentes, donde cada partición está conformada por uno o más procesos concurrentes (implementado por el sistema operativo de cada partición), que comparten el acceso a los recursos del procesador en función de las necesidades de la aplicación. El código de la partición puede ser una aplicación compilada para ser ejecutada sobre el hardware (máquina al desnudo), un sistema operativo en tiempo real (soporte en tiempo de ejecución) y sus aplicaciones, o un sistema operativo de propósito general y sus aplicaciones.

V. Porting de Android a XtratuM

Para aprovechar las características graficas de la interfaz de usuario de Android con una mayor capacidad y sea más sencillo de manejar todos los componentes integrados al sistema domótico, en un medio intuitivo para el usuario final y ya que XtratuM no soporta de forma nativa la arquitectura de los microprocesadores ARM que utiliza generalmente Android se decidió utilizar el porting de Android para la arquitectura x86 [12] el cual fue portado por la organización llamada Android-x86.org y este como hemos mencionado se portará con el XtratuM diseñado inicialmente para ´esta misma arquitectura.

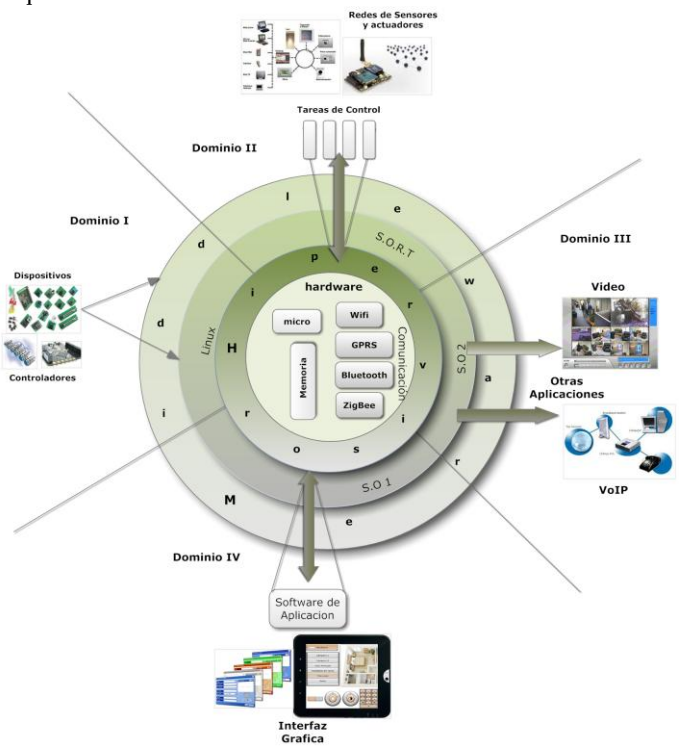

Fig. 3: Plataforma global del sistema domótico

Existe un parche para que Linux soporte el módulo de XtratuM, tomamos el parche, revisamos compatibilidad entre los componentes que necesitan (XtratuM y Android) para funcionar, que no haya conflictos entre ellos, después lo colocamos al kernel que utiliza Android, se corrigen los cambios de directorios, los cambios de variables, se revisa detalladamente la integridad de la funcionalidad de los cambios realizados.

La arquitectura general presentada para el porting de Android sobre XtratuM se muestra en la Fig. 4 como se puede observar, XtratuM toma los componentes de la base de Android que son los componentes del kernel de Linux, tomando el control entonces puede multiplexar las interrupciones y manejos de memorias dándole a Android la capacidad de ser un sistema operativo con características de tiempo real.

La problemática con la que nos encontramos ha sido los cambios existentes en el kernel de Linux implementado en Android y el parche de XtratuM para Linux, en los cuales debido a su versión hay diferencias en variables, estructuras, funciones, punteros así como cambio de nombres y directorios de archivos. Por lo cual hay que identificar esos cambios y hacer las correcciones debidas meticulosamente.

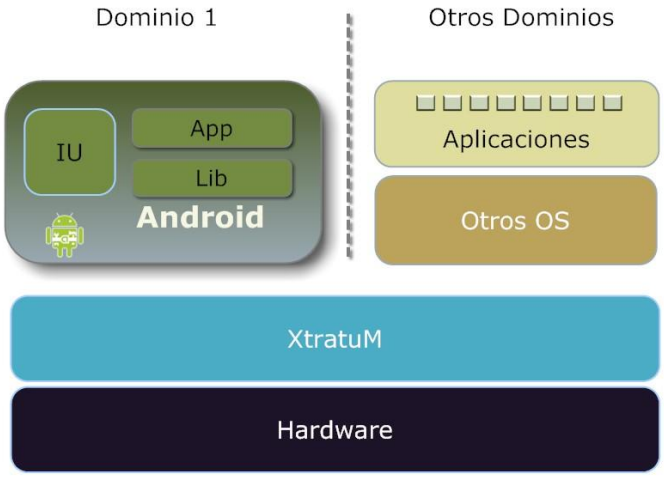

Fig. 4: Arquitectura global de Android portado a XtratuM

VI. Interfaz de Usuario para la Aplicación Domótica

Una interfaz de usuario (UI - User Interface por sus siglas en inglés) para la casa inteligente provee de un usuario el acceso a algunos o todos los dispositivos que están integrados en el hogar. Una vez que el usuario selecciona un dispositivo, su estado actual se puede mostrar, así como el menú de comandos disponibles. El equipo puede estar asociado a un determinado grupo de actividades, incluyendo la iluminación, entretenimiento, seguridad.

Fig. 5: Diseño de la Interfaz de Usuario General

Fig. 6: Diseño de la Interfaz de Usuario Ampliada

El acceso a cualquier dispositivo específico está sujeto a la autorización y la capacidad de las interfaces de usuario para controlar sus funciones. Los dispositivos de interfaz de usuario pueden tener varios niveles de sofisticación; por ejemplo, el dispositivo de interfaz puede reproducir música o facilitar el acceso a los sistemas de entretenimiento doméstico controlado por interfaces de usuario simple, mientras que las interfaces avanzados pueden controlar los teléfonos móviles, PDA's, pantallas táctiles y alta definición de interfaces multimedia, entre otros. Los dispositivos más avanzados tienen la ventaja de permitir un mejor uso de los gráficos y software, y puede presentar distintas pantallas.

Para esta aplicación se ha desarrollado una interfaz de usuario sobre el sistema operativo Android, con el SDK de este sistema en su versión 2.3. Como se mencionó esta interfaz consta de niveles de sofisticación, por lo tanto tendremos el desarrollo de aplicaciones locales como aplicaciones remotas con dispositivos móviles (celular, Tablet, Laptop, PDA, etc.).

Para la comunicación con los dispositivos móviles existe un servidor el cual tiene intercomunicación con los dispositivos de entorno como sensores y actuadores a través del protocolo IEEE 802.15.4 y con los móviles a través de WIFI, mediante un servidor web, el cual hace la conexión del exterior hacia el interior, para acceder a los datos del sistema y poder realizar funciones de control y automatización.

Mediante esta interfaz podemos visualizar representaciones de la casa donde está instalado el sistema domótico, además nos brinda la información necesaria para conocer el estado en el que se encuentra nuestro hogar y a partir de estos datos poder tomar decisiones, cambiar y controlar los recursos a

nuestra conveniencia en cuestión de seguridad, confort, ahorro energético, etc.

Se puede observar en la Fig. 5 que la interfaz contiene un plano de la casa, en este se puede acceder a cada área y ver cuál es su estado. Una vez dentro en la Fig. 6 se ve de manera gráfica una representación del lugar, así como información importante, tal como, temperatura, presencia, grado de luminosidad, estado de la alarma, entre otras. Además podemos hacer cambios como encender las luces, el aire acondicionado, monitorear el sistema de alarma entre otras aplicaciones, todo esto de manera remota.

En la Fig. 7 se puede aprecia la aplicación antes descrita funcionando sobre un teléfono inteligente (Smartphone), esto nos proporciona un nivel más alto de sofisticación. Esta aplicación podrá ser instalada sobre cualquier plataforma o dispositivo que tenga instalado Android pero solo para ser un mando a distancia, éste dispositivo no contendrá XtratuM por consiguiente la aplicación no será de tiempo real; se están realizando pruebas con celulares, tablets y con la conversión (porting) de este sistema para arquitectura x86, que en algunas aplicaciones tendrá el rol de servidor de datos. Esta aplicación es importante debido a que nos proporcionará la interacción del usuario con el sistema domótico de una manera fácil e intuitiva.

VII. Aplicación Dentro de la Domótica

Una casa inteligente típica consiste en sistemas heterogéneos, tales como los sistemas de acceso, controles de iluminación, gestión energética, seguridad y vigilancia digital, entre otros. Estos sistemas heterogéneos permiten a sus propietarios especificar las acciones y tareas que se llevarán a cabo en momentos específicos. Esto puede lograrse mediante el desarrollo de una arquitectura distribuida para dispositivos virtuales que se ejecutan en la parte superior de los niveles de aplicación y varios puntos de control en toda la casa, mientras que proporciona una interfaz de usuario para las personas que van administrar de forma remota.

En muchos casos, estas interfaces reflejan la interfaz basada en navegadores para el control y seguimiento. El alojamiento (hosting) virtual en módulos de terminales en el hogar proporciona acceso a aplicaciones de mayor capacidad de procesamiento y una mayor conectividad, que es más preferible que operen a nivel local con recursos limitados. Los algoritmos inteligentes en este tipo de sistemas heterogéneos pueden utilizar virtualizaciones de mayor poder de procesamiento, conectividad y tamaños de almacenamiento.

Uno de los dominios más populares para aplicaciones domóticas es el entretenimiento, y dentro de éste, uno de los mayores desafíos es la gestión del contenido. Los propietarios de viviendas están expuestos a una gran variedad de música, imágenes y medios de comunicación, gran parte de este contenido puede terminar en un solo dispositivo, en lugar de ser distribuidos entre los diferentes dispositivos que se utilizan de una forma más expandida entre un mayor número de miembros de la familia ubicados en diferentes lugares de la casa. La centralización de los datos crea dificultades para los habitantes de una casa porque necesitan cambiar de un dispositivo a otro para poder acceder a diferentes tipos de contenido. Usando la virtualización podría resolver este cuello de botella mediante la implementación de la virtualización de contenidos en una plataforma distribuida. Usando la

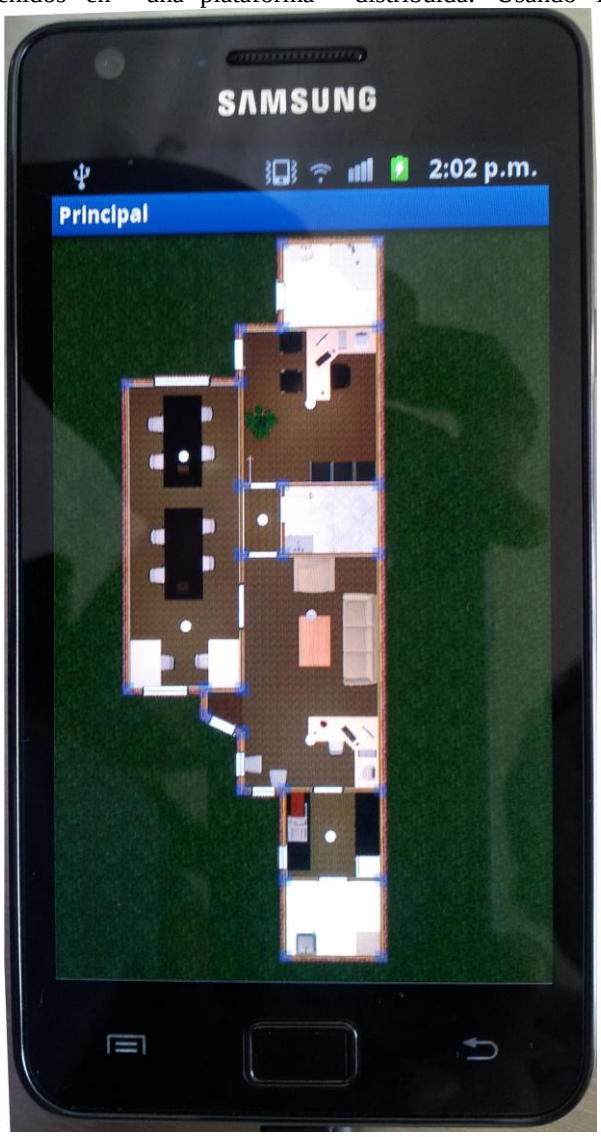

Fig. 7: Aplicación funcionando sobre Smartphone con Android

virtualización también se garantiza la organización, el etiquetado y el análisis de todos los contenidos de entretenimiento almacenados de una manera más eficiente y accesible inmediatamente. Por otro lado, la virtualización también prepara el camino para el reconocimiento de la actividad de las personas en función de su recuperación y utilización de contenidos específicos.

Por estas razones no es suficiente el API nativo de Android, se necesitan controlar las interrupciones para poder tener un mejor manejo de las aplicaciones distribuidas en la plataforma domótica. Una aplicación implementada sobre la plataforma interpretara los datos y a partir de su análisis podrá realizar una toma de decisiones las cuales comunicara a los actuadores los cuales tendrán la tarea de activar/desactivar dispositivos de salida. El hipervisor permite el cambio de contexto entre los sistemas operativos y las aplicaciones además de brindarles

protección y administrar los recursos del hardware.

VIII. Conclusiones

Lo antes visto para el proyecto es que Android como el SO Móvil que es, sirva de base para ser la interfaz gráfica de un ambiente domótico distribuido con XtratuM de hipervisor dando capacidades de ser un SO en tiempo real al controlar sus interrupciones de manera restringida, pudiendo ejecutar múltiples dominios (maquina virtuales) al mismo tiempo con otros Sistemas Operativos o aplicaciones.

La arquitectura de cómo sería el servidor dentro de la vivienda puede ser visto en general en la Fig. 4 este diseño intenta mostrar múltiples aplicaciones en tiempo real ejecutándose conjuntamente con el SO con una interfaz gráfica.

Una aplicación implementada sobre la plataforma interpretara los datos y a partir de su análisis podrá realizar una toma de decisiones las cuales comunicara a los actuadores los cuales tendrán la tarea de activar/desactivar dispositivos de salida. El hipervisor permite el cambio de contexto entre los sistemas operativos y las aplicaciones además de brindarles protección y administrar los recursos del hardware.

Pero esto no es en todo lo que se podría utilizar la implementación, teniendo un SO que su principal atractivo es la experiencia grafica hacia el usuario por su UI, siendo también un SO Móvil que necesita de pocos recursos computacionales para funcionar lo hacen un candidato para ser usado en múltiples aplicaciones embebidas de tiempo real, domótica, robótica, etc.

Referencias

[1] Erwin Scholtsch Embedded Systems - Introduction, European Research Consortium for Informatics and Mathematics, ERCIM News No 52, Jan- uary 2003.

[2] Georgio Butazzo. Research trends in real-time computing for embedded systems. ACM SIGBED Review. Volume 3, Issue 3 (July 2006). Special issue on major international initiatives on real-time and embedded systems Pages: 1 - 10. Year of Publication: 2006. ISSN: 1551-3688.

[3] Craig Hollabaugh. Embedded Linux, Hardware, Software and Interfacing. Ed. Addison-Wesley, February 2006. ISBN 0672322269.

[4] Chistopher Hallinan. Embedded Linux Primer Ed. Prentice Hall. 2007. ISBN 0-13-167984-8.

[5] Thomas A. Henzinger, Joseph Sifakis. The Discipline of Embedded Systems. Computer, vol. 40, no. 10, pp. 32-40, Oct. 2007.

[6] M. Masmano, I. Ripoll, and A. Crespo. An overview of the XtratuM nanokernel. In Proceedings of the Workshop on Operating Systems Platforms for Embedded Real-Time Applications (OSPERT), 2005.

[7] Tim Abels, Punnet Dhawan, and Balasubramanian Chandrasekaran. An Overview of Xen Virtualization. Dell Power Solutions, August 2005. www.dell.com/powersolutions.

[8]] Bryan Clark, Todd Deshane, Eli Dow, Stephen Evanchik, Matthew Finlayson, Jason Herne, and Jeanna Neefe Matthews. Xen and the art of repeated research. In USENIX Annual Technical Conference, FREENIX Track, pages 135-144, 2004.

[9] Inc. VMWare. Vmware workstation. http://www.vmware.com/.

[10] Sun VirtualBox. www.virtualbox.org.

[11] What is Android?, Junio 2011 , http://developer.android.com/guide/basics/what-is-android.html.

[12] Android-x86 Project - Run Android on Your PC, Noviembre 2011, http://www.android-x86.org/.

[13] A. Crespo, I. Ripoll, M. Masmano, P. Arberet y J. Metge, «XtratuM an Open Source Hypervisor for TSP Embedded Systems in Aerospace,» DAta Systems In Aerospace (DASIA), 2009, Istanbul, Turkey.

[14] OMG, «Data Distribution Service for Real-Time Systems version 1.2,» OMG Technical Document, Jan. 2007.

[15] A. J. Vico, "La columna 80." [Online]. Noviembre 2011, http://columna80.wordpress.com/