“DESARROLLO DE UN GESTOR DE DISPOSITIVOS UPNP (UNIVERSAL PLUG AND PLAY)”

HECTOR GUILLERMO DUEÑAS ROJAS

LUIS FERNANDO GUTIERREZ PRADO

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE CIENCIAS FISICO MECANICAS

ESCUELA DE INGENIERIA DE SISTEMAS E INFORMATICA

BUCARAMANGA

2010

“DESARROLLO DE UN GESTOR DE DISPOSITIVOS UPNP (UNIVERSAL PLUG AND PLAY)”

HECTOR GUILLERMO DUEÑAS ROJAS

LUIS FERNANDO GUTIERREZ PRADO

Proyecto de Grado presentado como requisito

Para optar al título de ingeniería de sistemas

DIRECTOR

Ph.D. Sergio Castillo Castelblanco

CODIRECTOR

Ing. Ariel Yezid Villareal

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE CIENCIAS FISICO MECANICAS

ESCUELA DE INGENIERIA DE SISTEMAS E INFORMATICA

BUCARAMANGA

2010

Este proyecto está dedicado a mis padres y hermana

Quienes con su apoyo y cariño contribuyeron a

Que este proyecto fuera realidad

Gracias a Dios, por permitirnos llegar hasta esta gran

Meta de ser Profesional, Muchas gracias.

HECTOR GUILLERMO DUEÑAS ROJAS

Este proyecto es dedicado a nuestras familias
Que nos han apoyado durante estos años en los
Que nos hemos preparado para convertirnos en
Ingenieros de sistemas; ellos han tenido la paciencia
Y la fortaleza para guiarnos por el buen camino
Que nos llevo a culminar esta etapa de nuestras vidas;
Ellos han crecido al lado de nosotros y han compartido
Lo bueno y lo malo que nos ha brindado la vida,
Ahora ha llegado el momento de contribuir a sus
Esfuerzos, y que se sientan contentos y orgullosos de haber
Logrado junto con nosotros ese anhelo de ser profesional;
Gracias Dios, gracias a todos ellos.

LUIS FERNANDO GUTIERREZ

AGRADECIMIENTOS

Los más sinceros agradecimientos a:

Doctor Sergio Castillo Castellano, Ingeniero de Sistemas, Ph.D. Profesor de la Escuela de Ingeniería de Sistemas e Informática y Director del proyecto. Por su constante apoyo, colaboración y asesoría en el desarrollo de este proyecto

TABLA DE CONTENIDO

Pág.

LISTADO DE FIGURAS

Pág.

LISTADO DE ANEXOS

RESUMEN

Titulo
*DESARROLLO DE UN GESTOR DE DISPOSITIVOS UPnP (Universal Plug and Play)**

Autores
HECTOR GUILLERMO DUEÑAS ROJAS
LUIS FERNANDO GUTIERREZ PRADO**

Palabras Claves:
Protocolo UPnP, Computación ubicua, Red inalámbrica, Domotica, Fórum UPnP, Librería.

Descripción:

El principal objetivo del presente proyecto de grado consistió en diseñar y desarrollar una herramienta software que permite detectar, reconocer, enumerar, manipular y presentar los servicios y propiedades de los diferentes dispositivos capaces de comunicarse mediante el protocolo UPnP presentes en el interior de una red domestica, cuya configuración puede ser inalámbrica o cableada. Esta herramienta permite que un usuario pueda controlar por medio del intercambio de mensajes de control los posibles estados del dispositivo de su interés, como la intensidad de la luz, el volumen de un reproductor de audio y video entre otros, este tipo de funcionalidad hace que el protocolo UPnP resulte muy adecuado para la implementación de la Domotica, ciencia que pretende diseñar recintos inteligentes que necesiten de la mínima intervención del hombre para su configuración y funcionamiento, siendo esta la rama más avanzada de la computación ubicua.

El Gestor de Dispositivos UPnP, como ha sido nombrada esta aplicación fue creado a partir de la librería upnp.dll proporcionada por el creador del protocolo UPnP la empresa desarrolladora de software Microsoft, la cual incluye este protocolo en todas sus versiones de sistemas operativos desde Windows XP hasta el actual Windows Siete. También se siguieron las especificaciones técnicas establecidas por el Fórum UPnP, grupo encargado de impulsar el uso del protocolo UPnP como estándar en las comunicaciones del hogar inteligente o Domotico.

En cuanto a la plataforma de desarrollo, el proyecto fue realizado en el entorno del sistema operativo Windows XP ServiPack 3 y utilizando la tecnología Visual Studio C# junto con el lenguaje C++. El gestor consta de dos módulos, el modulo encargado de la búsqueda y control el cual es una librería desarrollada en C++, y el modulo de presentación de resultados e interfaz de usuario desarrollado en C# implementando el modulo de búsqueda de C++.

---

*Proyecto de grado modalidad investigación.

**Facultad de Ingenierías Fisico-Mecanicas, Escuela de Ingeniería de Sistemas e Informática.
Director Ph.D. Sergio Castillo Castelblanco – Codirector: Ing. Ariel Yezid Villareal.

ABSTRACT

Title
DEVELOPMENT OF A DEVICE MANAGER UPnP (Universal Plug and Play)*

Authors
DUEÑAS GUILLERMO HECTOR ROJAS
PRADO LUIS FERNANDO GUTIERREZ**

Keywords:
UPnP protocol, Ubiquitous computing, wireless networking Home Automation, UPnP Forum, Library.

Description:

The main objective of this graduation project was to design and develop a software tool to detect, recognize, list, handling and presenting the services and properties of different devices capable of communicating via the UPnP protocol present in the interior of a home network, whose configuration can be wireless or wired. This tool allows a user to control through the exchange of control messages the possible states of the device of interest, such as light intensity, the volume of an audio and video among others, this type of functionality makes UPnP protocol is well suited for the implementation of the Home Automation, science that aims to design intelligent enclosures that require minimal human intervention for setup and operation, this being the most advanced branch of ubiquitous computing.

The UPnP Device Manager, and has been named this application was created from the library Upnp.dll provided by the creator of the UPnP protocol software developer Microsoft, which includes this protocol in all versions of Windows operating systems from Windows XP to the current seven. Also continued the technical specifications established by the UPnP Forum, a group responsible for promoting the use of UPnP protocol standard on smart home communications and home automation.

As a development platform, the project was implemented in the environment of Windows XP operating system ServiPack 3 and using the Visual Studio C# technology together with the C++ language. The manager consists of two modules, the module responsible for search and control which is a library developed in C++, and results presentation module and user interface developed in C# implementing the search module C++.

---

* Proposed research degree mode.

** Faculty of Engineering Physics-Mechanical, School of Engineering Systems and Informatics. Director Ph.D. Sergio Castillo Castelblanco - Co-Director: Mr. Ariel Yezid Villareal.

GLOSARIO

API: Interfaz de Programación de Aplicaciones o API es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción. Son usados generalmente en las bibliotecas.

Computación Ubicua: Es el acceso a gran cantidad de información y procesamiento de la misma independientemente de la ubicación de los usuarios. Esto implica la existencia de una gran cantidad de elementos de computación disponibles en un determinado entorno físico y constituido en redes.

DCP: Protocolo de control de dispositivos (Device Control Protocol).

DCPD: Documento del protocolo de control de dispositivos (Device Control Protocol Document).

DHCP: Protocolo de Configuración de Host Dinámico. Es un protocolo que permite que un equipo conectado a una red pueda obtener su configuración (principalmente, su configuración de red) en forma *dinámica* (es decir, sin intervención particular). Sólo tiene que especificarle al equipo, mediante DHCP, que encuentre una dirección IP de manera independiente. El objetivo principal es simplificar la administración de la red.

Dirección IP: Es un número que identifica un ordenador dentro de una red que utilice el protocolo IP.

Domotica: conjunto de servicios de la vivienda garantizado por sistemas que realizan varias funciones, los cuales pueden estar conectados entre sí y a redes interiores y exteriores de comunicación. Gracias a ello se obtiene un notable ahorro de energía, una eficaz gestión técnica de la vivienda, una buena comunicación con el exterior y un alto nivel de seguridad".

Interfaz o Interface: Conexión e interacción entre hardware, software y el usuario.

IP: El Protocolo de Internet es un protocolo usado tanto por el origen como por el destino para la comunicación de datos a través de una red local o Internet.

IPv4: Versión 4 del protocolo de Internet (IP version 4).

Protocolo: Es un conjunto de reglas que especifican el intercambio de datos u órdenes durante la comunicación entre sistemas de ordenadores o redes.

RPC: Llamada de procedimiento remoto (Remote Procedure Call).

SSDP: Protocolo Simple de Descubrimiento de Servicios es un protocolo que sirve para la búsqueda de dispositivos UPnP en una red.

SDK: Kit de desarrollo estándar (Standard Development Kit).

TCP: Protocolo de control de transmisión (Transmission Control Protocol).

TTL: Tiempo de vida (Time-To-Live).

UDP: Protocolo de datagramas de usuario (User Datagram Protocol).

UML: Lenguaje unificado de modelado (Unified Model Language).

UPnP: Conectar y usar universal (*Universal Plug and Play*) .

URI: Identificador de recurso uniforme (Uniform Resource Identifier).

URL: Localizador Uniforme de Recursos. Forma de organizar la información en la web. Una URL es una dirección que permite acceder a un archivo o recurso como pueden ser páginas HTML, php, asp, o archivos gif, jpg, etc. Se trata de una cadena de caracteres que identifica cada recurso disponible en la WWW.

USN: Nombre de serie único (Unique Serial Name).

UUID: Identificador único universal (Universal Unique IDentifier).

XML: Acrónimo del inglés eXtensible Markup Language (lenguaje de marcado ampliable o extensible) desarrollado por el World Wide Web Consortium (W3C).

INTRODUCCION

La iniciativa de llevar a cabo el presente proyecto surgió como resultado de la observación realizada en el actual medio tecnológico y confirmar que la sociedad cada vez más se inclina hacia el uso de la tecnología en el hogar, sin embargo diariamente casi todos los dispositivos electrónicos son conectados a través de cables que en ocasiones representan dificultades y quizás no sea estético, por suerte han venido surgiendo redes inalámbricas que representan más espacio disponible en el área de trabajo; también se aprecio que se ha presentado la necesidad de controlar los dispositivos electrónicos que se encuentren a disposición en el área de trabajo, esto tiene una similitud con lo que se conoce como hogar inteligente o domótico , sin embargo el proyecto va un poco más allá de activar un dispositivo porque también es posible intercambiar datos con un dispositivo de acuerdo a la tecnología que se maneje, de hecho lo que se pretende realizar con este trabajo es enfocar a futuro posibles trabajos en los cuales se puedan realizar sincronización entre los dispositivos para poder realizar tareas complejas que requieran de varios dispositivos que trabajen en determinado momento.

El protocolo UPnP está siendo utilizado cada día por más empresas a nivel mundial, con el fin de globalizar este servicio, y así ofrecer a las personas una forma más fácil de manejar la vida diaria en la oficina, en la casa, en un restaurante, en fin donde se pueda implementar esta tecnología.

1. PLANTEAMIENTO DEL PROBLEMA

1.1. ANTECEDENTES Y SITUACION DEL PROBLEMA

En la actualidad se puede observar que los hogares del futuro tienden a relacionar todos los dispositivos electrónicos existentes en su interior, como lo son los ordenadores, las consolas de juegos, los reproductores de audio y video, impresoras entre otros, de forma automática sin la necesidad de utilizar cables para su conexión. De esta manera se puede controlar cualquier dispositivo de una red sin tener que estar delante de él, junto con el abanico de posibilidades que esto abre. Por ejemplo se puede reproducir en el salón la película que se acaba de descargar en el computador de escritorio, sin tener que ir a grabarla en una memoria o DVD, también se podrá controlar lo que están viendo los niños en su habitación sin tener que ir hasta allí.

Para lograr tal grado de interacción surge como intermediario el protocolo UPnP[1] de Microsoft, el cual busca convertirse en un estándar de comunicaciones, para el Hogar Digital, sin embargo el UPnP quizás no solo se aplique al hogar sino también en las empresas o fabricas en donde se necesitan controlar procesos coordinados y automáticos, de hecho lo ideal es que a través de un dispositivo como un celular se puedan enviar ordenes a un punto de control u ordenador.

De hecho en la actualidad los celulares están empezando a implementar el protocolo UPnP.

[1] UPnP: Universal Plug and Play.

El protocolo UPnP es capaz de implementar las fases necesarias mediante el uso de otros protocolos especializados con el fin de lograr que un dispositivo u ordenador conectado a una red pueda intercambiar información con el resto de dispositivos conectados y determinar de esta forma que funcionalidad tienen, para posibilitar el mayor aprovechamiento de recursos. El protocolo UPnP ha sido diseñado de forma que sea independiente del fabricante, sistema operativo, del lenguaje de programación, de cada dispositivo u ordenador, y del medio físico usado para implementar la red, así encontraremos redes inalámbricas o cableadas con distintas topologías compartiendo un mismo lenguaje.

En la siguiente grafica se puede apreciar un ejemplo de red domestica comunicada mediante el protocolo UPnP.

Figura 1. Red Inalámbrica implementando UPnP para comunicarse

FUENTE: AUTORES basados en la especificación del Foro UPnP.

Este protocolo es capaz de descubrir cuando se conecta un nuevo equipo o dispositivo a la red, al cual se lee asigna una dirección IP, un nombre lógico, además de informar a los demás nodos de sus funciones y capacidad de procesamiento, e informarle, a su vez al dispositivo, de las funciones y prestaciones de los demás nodos. De esta forma, el usuario no tiene que preocuparse de configurar la red ni de perder el tiempo instalando drivers o controladores de dispositivos, el protocolo UPnP se encarga de todos estos procesos cada vez que se conecta o se desconecta un equipo. Y además optimiza en todo momento la configuración de los equipos.

En este escenario de ideas sobre el protocolo UPnP y sus beneficios, se observa que el protocolo UPnP se encarga de todo lo que se refiere a la conexión y desconexión de dispositivos pero no ofrece el desarrollo de una actividad en particular. Ya que la idea es lograr la interacción de dispositivos y servicios con el mínimo de esfuerzo para llevar a cabo una determinada tarea, se hace necesario crear algo que de orden a la creciente interconexión y desconexión de dispositivos y que permita al usuario realizar la actividad deseada, como por ejemplo imprimir un documento o fotografía , acceder a una cámara de video, reproducir audio o video entre otras actividades, por lo que para solucionar esta problemática se hace necesario el desarrollo de un centro de mando que permita al usuario tener el control sobre las aplicaciones a desarrollar, a este centro de mando es el que se ha denominado Gestor de Dispositivos UPnP.

1.2. **OBJETIVOS**

1.2.1. OBJETIVO GENERAL

Desarrollar un gestor de dispositivos UPnP que permita el descubrimiento, la administración y la colaboración entre dispositivos UPnP.

1.2.2. Objetivos Específicos

1. Realizar la especificación de requisitos del prototipo de gestor de dispositivos UPnP.
2. Diseñar e implementar el modulo de detección de dispositivos que permita:
   - Detectar el ingreso de un nuevo dispositivo UPnP a la red mediante el uso del protocolo SSDP[2].
   - Enumerar los dispositivos, servicios y aplicaciones presentes en el interior de la red.
   - Permitir la colaboración entre dispositivos UPnP mediante el intercambio de mensajes de control y de datos.
3. Diseñar e implementar el modulo de visualización encargado de:
   - Crear una interfaz de aplicación capaz de encontrar, integrar y controlar los dispositivos UPnP.
   - Obtener y visualizar una breve descripción de cada dispositivo UPnP conectado en red.

[2] SSDP: Protocolo Simple De Descubrimiento de Microsoft.

4. Realizar pruebas de integración, depuración y corrección de errores a cada uno de los módulos implementados.

1.3. JUSTIFICACION

Las grandes multinacionales desarrolladoras de tecnología debido a la competencia que existe en el mercado tienden a agregar a sus dispositivos electrónicos todos los servicios que le sean posibles para hacerlos superiores frente a los dispositivos de las empresas rivales, y por supuesto los servicios de UPnP no serán la excepción, de hecho ya existen algunos productos en el mercado como celulares, routers, impresoras, consolas de videojuegos entre otros que implementan este servicio.

En nuestro país la mayoría de la población ha desarrollado una adicción al consumo de las nuevas tecnologías, y si además esos productos incluyen servicios UPnP entonces las personas tendrán la posibilidad de conocer y aprovechar los beneficios ofrecidos por este protocolo y además podrán utilizar sus aplicaciones a través de un gestor de dispositivos UPnP; pues si estas personas poseen en el hogar un computador y varios dispositivos compatibles con el protocolo UPnP, tiene la posibilidad de usarlos de una forma coordinada y sencilla sin la necesidad de saber nada sobre las conexiones en red, aprovechando esto para poder desarrollar cierto tipo de tareas que requiere del trabajo en red.

La idea del proyecto es crear un software gestor de dispositivos UPnP que permita sacar provecho de algunos de los beneficios presentados por el protocolo UPnP, ofreciéndole al usuario una herramienta de uso intuitivo que le facilite el intercambio de mensajes de control y de colaboración entre un conjunto de dispositivos UPnP que se conectan y desconectan de forma automática a la red.

1.3.1. Impacto

El desarrollo de ésta investigación pretende provocar un impacto en dos importantes aspectos: social y técnico así:

Social:

Con la realización del presente proyecto se quiere dar a conocer a los usuarios en general el desarrollo de una propuesta por lograr un estándar en conexiones a red como lo es el protocolo UPnP, con el cual se pretende que para conectarse a una red solo sea necesario adquirir un dispositivo UPnP encenderlo y listo, empezar a trabajar con él, sin la necesidad de instalar drivers y sin importar el fabricante del mismo.

La idea de configurar automáticamente la conexión en red pretende llevar a todo tipo de usuarios desde los más expertos hasta los más novatos a trabajar en equipo no solamente con los dispositivos a su alcance si no también en comunidad con otros usuarios.

Técnico:

Este proyecto presenta una investigación sobre la convergencia de tecnologías es decir dispositivos y servicios interconectados, lo que permite a dispositivos diferentes trabajar en conjunto sin ninguna complicación de compatibilidad para el desarrollo de muchas actividades.

1.3.2. Viabilidad

Las herramientas necesarias para el desarrollo y funcionamiento de la aplicación son las siguientes:

El entorno utilizado para el desarrollo de la aplicación será Microsoft Visual Studio .Net un entorno de desarrollo integrado para sistemas Windows. El cual soporta varios lenguajes de programación como J#, C# y C++ (utilizados en nuestro caso), ASP .NET, Visual Basic .NET, utilizando las licencias software de las que dispone la universidad.

En cuanto a las herramientas hardware, tanto para la programación como para las pruebas del gestor solo se requiere de un computador con tarjeta de red inalámbrica funcionalidad incluida en cualquier computador portátil de hoy en día. Por otra parte los dispositivos UPnP necesarios para el desarrollo y pruebas de funcionamiento del gestor serán implementados mediante dispositivos UPnP software los cuales se ejecutaran sobre computadores portátiles y de escritorio los cuales estarán conectados de manera inalámbrica para conformar la red UPnP domestica de prueba.

1.4. ALCANCES

El gestor de dispositivos UPnP es también conocido como “punto de control” y se utiliza para crear aplicaciones capaces de descubrir y controlar los servicios ofrecidos por los dispositivos UPnP. Los Dispositivos UPnP se pueden conectar a cualquier red a la que tienen acceso ya sea esta cableada o inalámbrica o una combinación de las dos. Además estos dispositivos pueden ser de dos tipos, dispositivos hardware como un router o una impresora o dispositivos software como un reproductor de música o una bombilla binaria.

El gestor UPnP dispone de todos los mecanismos necesarios para encontrar y controlar los dispositivos UPnP, así como los mecanismos relacionados con la notificación sobre eventos generados por el dispositivo.

La interfaz del punto de control puede ser utilizada en aplicaciones escritas en Visual C++, Visual Basic o Visual C# para Microsoft Windows.

2. ESTADO DEL ARTE

2.1. COMPUTACION UBICUA

2.1.1. Definición

Figura 2. Escenario de la Computación Ubicua en el mundo

FUENTE: http://sociedadubicua.blogspot.com/2009_07_19_archive.html

Existe un uso generalizado del PC, que permite una computación democrática, donde cada persona posee y controla un ordenador de moderado costo, que progresivamente ha ido haciéndose más sencillo de manejar, de forma que personas sin preparación técnica, pueden hoy personalizar y adaptar a sus necesidades aplicaciones, ideadas para el gran consumo, que cada vez son más intuitivas y sencilla de manejar.

Se entiende por **computación ubicua** (ubicomp) la integración de la informática en el entorno de la persona, de forma que los ordenadores no se perciban como objetos diferenciados. Esta disciplina se conoce en inglés por otros términos como “Pervasive Computing, Calm Technology, Things That Think y Everyware”. Desde hace unos años también se denomina inteligencia ambiental.

Sus promotores tienen como objetivo insertar dispositivos inteligentes tanto en el entorno como en aparatos de uso diario para que las personas puedan interactuar con ellos de una manera natural y desinhibida en todo tipo de situaciones y circunstancias.

Ahora parece que entramos en el albor de una nueva era, la de la Computación ubicua caracterizada porque a cada persona actuará sobre una multitud de dispositivos programables. Dada su abundancia, necesariamente han de ser manejados con ninguno o mínimo esfuerzo, siendo en la mayor parte de los casos la interactividad entre el sujeto y la máquina absolutamente transparente, pues bastará con la maquina perciba su presencia para que interactúe con él, sin que en la mayoría de los casos el usuario tenga que hacer nada de forma consciente para ordenarlos, por lo que en la mayoría de los casos ni se percatara de su presencia. [21]

2.1.2. Campos de aplicación

New Songdo City es una ciudad ubicua (o ciudad-U) que se está construyendo en una isla frente a la ciudad de Inchon, a 60 kilómetros al oeste de Seúl (Corea del Sur). En una superficie de 680 hectáreas se construye esta ciudad en que todos los sistemas de información estarán interconectados y las computadoras estarán integradas a las viviendas, las calles y los edificios de oficinas. Con un presupuesto de 25 mil millones de dólares, la ciudad está siendo emplazada como

Zona Económica Libre. Se espera que la ciudad esté terminada para el 2014 y albergue a 65 mil personas, de las cuales unas 30 mil trabajarán ahí mismo.

A la cabeza, como en otros mucho campos se encuentra el MIT (Massachusetts Institute of Technology) que desde su laboratorio Media Lab impulsa la empresa Ambient devices fabricante objetos con esta filosofía, también la multinacional Philips dedica esfuerzos en esta área, que por ahora han dado como resultado una línea de productos llamada Smart Connections. La filosofía de todos estos "objetos inteligentes" Estos objetos de uso cotidiano tienen las cualidades interactivas "suaves" y no intrusivas que caracterizan a la Inteligencia ambiental.

Centrándonos en los productos "serios" desarrollados en España, no podemos dejar de nombrar al grupo de investigadores del Grupo CHICO (Computer-Human Interaction and Collaboration) de la E. S. de Informática de la Universidad de Castilla La Mancha, que trabaja en el sistema computación ubicua AULA, para la enseñanza del idioma, a través de ejercicios de escritura.

Dependiente de la Universidad de Lleida, el grupo GRIHO (Grupo de Investigación en Interacción Persona Ordenador), están desarrollando un guía turístico para los visitantes del Montsec, que tiene en sus explicaciones en cuenta el posicionamiento geográfico y el perfil del visitante, su nivel cultural, edad, preferencias, intereses, etc. así como el historial de las visitas realizadas. Un prototipo de lo que en el futuro pueden ser los guías digitales que nos acompañen en las visitas de cualquier bien del patrimonio artístico o museístico de cualquier punto del planeta.

Ikerland empresa perteneciente a las Cooperativas de Mondragón, con el apoyo del gobierno Vasco, realiza en este momento varios proyectos relacionados con esta tecnología. El Var Trainer, un simulador para el entrenamiento en maquinaria de construcción, el Space4u, relacionado con la Inteligencia Ambiental de centros de convenciones y similares. Se espera que esta organización consiga buen fruto de este esfuerzo por ser punto de unión de su experiencia demostrada en las

áreas de construcción mecánica, tanto de equipos pesados como miniaturizados, en el de la tecnología digital y en su especialización en domótica y edificios inteligentes. [21]

2.1.3. Ventajas.

La computación ubicua emplea un conjunto de pequeños dispositivos móviles interconectados inalámbricamente entre sí para la consecución de una tarea común. La diferencia con otros esquemas de computación radica en que esta cooperación se realiza mediante una muy baja interacción con el usuario y de una forma casi transparente a éste.

A continuación se describe el estado actual y futuro de la tecnología en este campo, haciendo especial énfasis en la relación de la computación ubicua con otras disciplinas tecnológicas como la domótica, la computación móvil o los dispositivos vestibles (computación ubicua dialnet).

La computación Ubicua Incorpora tres nuevos conceptos:

• **Uso eficaz de espacios "perspicaces"**. Se basa, en la detección del estado de un individuo y de sus necesidades, deducidas de dicho estado, ya sea en la oficina, sala de reuniones, clase, domicilio, coche, etc. El concepto ha sufrido una evolución respecto de la idea original. Al comienzo la arquitectura del sistema era más cercana a la de un sistema distribuido, un servidor central, gobernando el funcionamiento de un conjunto de dispositivos/clientes. Actualmente, los dispositivos tienen mayor perspicacia individualmente. El espacio perspicaz surge cuando varios de estos dispositivos coinciden en el mismo espacio físico e interactúa colaborativamente La domótica, computación ubicua en el domicilio para dar soporte a los individuos que se encuentren en él. , es la aplicación más popular.

• **Invisibilidad**. Actualmente, se está lejos de la propiedad expuesta por Weiser para los sistemas ubicuos, la completa desaparición de la tecnología de la consciencia del usuario. Una buena aproximación es tener presente, en el diseño de estos sistemas, la idea de mínima distracción del usuario. La invisibilidad va a requerir del cambio drástico en el tipo de interfaces que nos comunican con los computadores. Reconocimiento de voz y de gestos, comprensión del lenguaje natural y del texto manuscrito, en la dirección hombre-máquina y en el sentido contrario, síntesis de lenguaje hablado y escrito y de representaciones gráficas.

• **Escalabilidad local**. Una idea que sorprende, al introducirnos es este campo por primera vez, es el antagonismo existente entre la filosofía de Internet y la de la computación ubicua. A priori, es natural pensar en la computación ubicua como un trasvase de la filosofía de Internet; cualquier servicio de computación, en cualquier sitio. Nada más alejado de la realidad. El concepto de localidad de servicios en computación ubicua es fundamental frente a la universalidad de servicios de Internet. Los usuarios disponen de capacidades asociadas al contexto en el que se encuentran. Por ejemplo, carece de sentido de que las aplicaciones domóticas situadas en el domicilio particular tengan que estar escrutando las necesidades del usuario que se encuentra trabajando en ese momento en la oficina. Al igual que la mayoría de las interacciones en la naturaleza, la proporcionada por estos sistemas, decrece con la distancia al usuario.

2.2. UNIVERSAL PLUG AND PLAY (UPnP)

En los próximos subapartados se estudiará la tecnología UPnP. Se expondrán las bases conceptuales de ésta tecnología para posteriormente describir la arquitectura que define, así como los protocolos sobre los que está basada y las

distintas fases de una comunicación UPnP y se enumerarán las limitaciones que presenta. Finalmente se justificará su uso en el presente proyecto.

2.2.1. Introducción

UPnP, de las siglas en inglés *Universal Plug and Play*, es un protocolo de transmisión de datos punto a punto para la comunicación entre aplicaciones. Define una arquitectura software abierta y distribuida, que proporciona una forma de conectividad entre distintos equipos pertenecientes a una red, permitiendo el intercambio de información y datos entre los mismos.

Surge a partir del modelo *Plug and Play* para la conexión de periféricos al PC (*Personal Computer*), cuya máxima, “enchufar y listo para usar”, es extendida a dispositivos inteligentes, dispositivos móviles, PCs, electrónica de consumo, y aplicaciones software.

Usando el modelo UPnP, un dispositivo puede agregarse dinámicamente a una red, obtener una dirección IP, anunciar sus servicios, y saber de la existencia de otros dispositivos, todo esto de forma automática y transparente para el usuario final.

Varios son los escenarios en los que se puede implementar. Ejemplos de estos son la automatización del hogar, impresoras en red, electrodomésticos para la cocina, o el entretenimiento digital.

Según el Foro UPnP esta tecnología aporta beneficios tales como:

- **Independencia de medios (tipos de redes) y dispositivos:** Puede funcionar sobre cualquier medio que soporte el protocolo IP, incluyendo líneas telefónicas o eléctricas, Ethernet, FireWare, Infrarrojos (IR), radio (RF): Wi-Fi, Bluetooh, etc. Esto hace que su uso sea apropiado para la Domótica.

- **Independencia de la plataforma y del lenguaje de programación:** Se puede usar sobre cualquier sistema operativo y utilizando cualquier lenguaje de programación para crear productos UPnP. De hecho, UPnP no especifica ninguna API (*Aplication Programming Interface*) para aplicaciones, dando libertad a los desarrolladores y fabricantes de utilizar sistemas operativos y capacidades acordes a las necesidades de sus clientes.

- **Tecnologías basadas en Internet:** Está desarrollada sobre protocolos comunes de Internet tales como IP (*Internet Protocol*), TCP (*Transmission Control Protocol*), UDP (*User Datagram Protocol*), HTTP (*Hypertext Protocol*) y XML (*Extensible Markup Language*)

- **Interfaz de usuario de control del dispositivo:** La arquitectura UPnP permite presentar una interfaz de usuario en un explorador web a través de la cual un usuario final podría, dependiendo de las capacidades del dispositivo, controlar a éste en remoto.

- **Control por programa de la aplicación:** Un programa puede controlar directamente un dispositivo UPnP sin necesidad de que éste posea un interfaz de usuario. Este control puede basarse en la información descubierta sobre el dispositivo o a través de un conocimiento a priori del tipo de dispositivo.

- **Protocolos base comunes:** Esta tecnología se fundamenta en protocolos y procedimientos comunes, con los que los desarrolladores de aplicaciones y dispositivos están de acuerdo, asegurando la interoperabilidad entre los múltiples dispositivos existentes en el mercado.

- **Extensibilidad:** Los fabricantes o desarrolladores pueden introducir servicios de valor añadido a sus productos UPnP, sobre una arquitectura básica común.

El Foro UPnP es una asociación de compañías y personas relacionadas con la industria del sector tecnológico, formada en Octubre de 1999, cuyo propósito es promover el desarrollo de dispositivos de fácil conexión y simplificar su implementación en redes del hogar y entornos empresariales. Esto lo logra definiendo y publicando descripciones de Dispositivos y Servicios UPnP, originalmente denominadas DCPs (*Device Control Protocols*), de acuerdo a una arquitectura común de dispositivo impulsada por Microsoft.

2.2.2. Componentes de una red UPnP

La arquitectura UPnP define dos nodos principales de comunicación: los Puntos de Control y los Dispositivos.

2.2.2.1. Dispositivos y Servicios

Un dispositivo UPnP puede ser tanto un dispositivo real (hardware), como una aplicación software. Se trata de un contenedor de servicios y/o otros dispositivos. Estos servicios y dispositivos embebidos tienen como objetivo definir la funcionalidad del dispositivo que los contiene.

Poniendo como ejemplo de dispositivo contenedor un televisor LCD con reproductor DVD integrado. Los servicios que contiene podrían ser: un servicio de control de la visualización, un servicio de sintonización, y un servicio de reloj. A su vez tiene un dispositivo embebido, el reproductor DVD, con sus propios servicios asociados, como por ejemplo: un servicio de control de la reproducción, un servicio de afinamiento, etc. El conjunto de los servicios y dispositivos integrados define las características funcionales del televisor combo.

Toda esta información se recoge en un documento XML de descripción, DCPD (*Device control Protocol Document*), que el propio dispositivo debe alojar.

Los servicios conforman la unidad más pequeña de control en una red UPnP. Proporciona un conjunto de acciones y modela su estado con variables de estado. Siguiendo con el ejemplo del televisor LCD, una variable de estado del servicio de reloj podría ser la hora actual, y una acción sería obtener dicha hora.

De forma análoga a la descripción del dispositivo, esta información queda reflejada en un documento XML de descripción del servicio denominado SCPD (*Service Control Protocol Definition*).

Además, los cambios de estas variables de estado podrían generar eventos que las entidades subscritas para escucharlos, comúnmente los Puntos de Control, pueden utilizar para mantener actualizada la información de estado del dispositivo controlado.

2.2.2.2. Puntos de Control

Los puntos de control UPnP son controladores de dispositivos UPnP, capaces de descubrir y comunicarse con ellos. Después de escuchar los mensajes de descubrimiento de los dispositivos, pueden obtener la descripción de los mismos con la lista de servicios asociados y sus correspondientes descriptores, invocar acciones para controlar su comportamiento, e incluso subscribirse a la fuente de eventos de estos para poder enterarse de los cambios en sus variables de estado.

En la siguiente figura se muestra la relación entre los servicios de los dispositivos y los puntos de control:

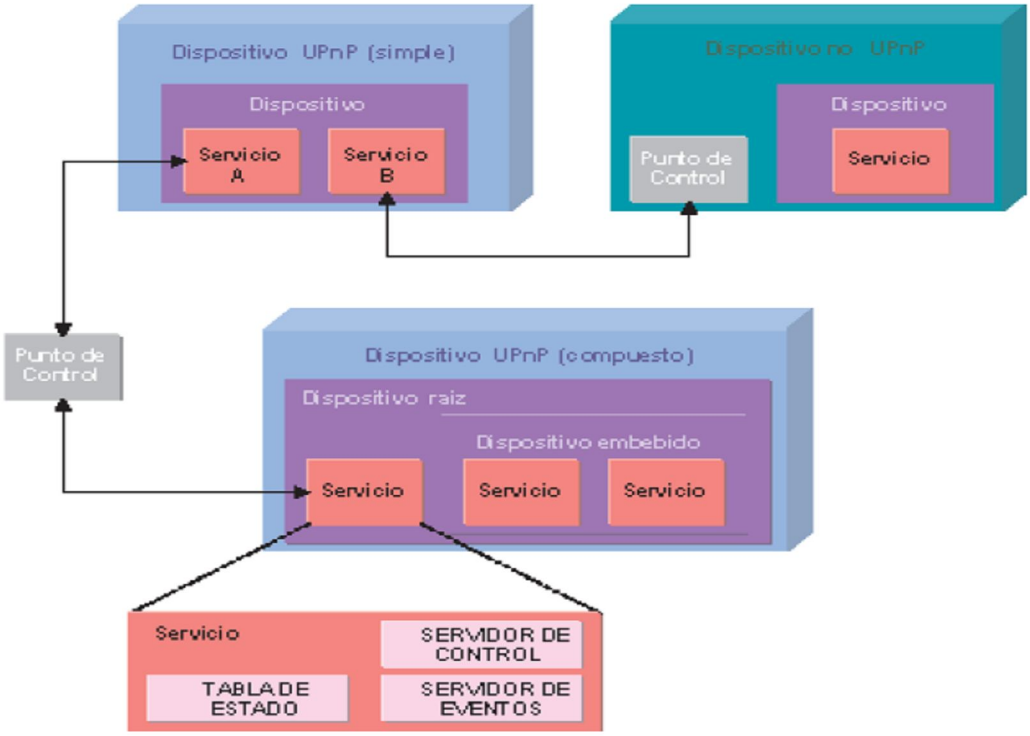


Figura 3. Esquema Puntos de Control, Dispositivos y Servicios

FUENTE: http://www.fundacionorange.es/areas/historico/pdf/4.pdf

2.2.2.3. Puentes (Bridge)

Los puentes UPnP, son entidades de enlace entre la red UPnP y dispositivos que no pueden pertenecer a ésta por carecer de recursos o por no soportar protocolos como TCP/IP, HTTP o XML. Un ejemplo de escenario con un UPnP *Bridge* sería el definido en la siguiente figura.

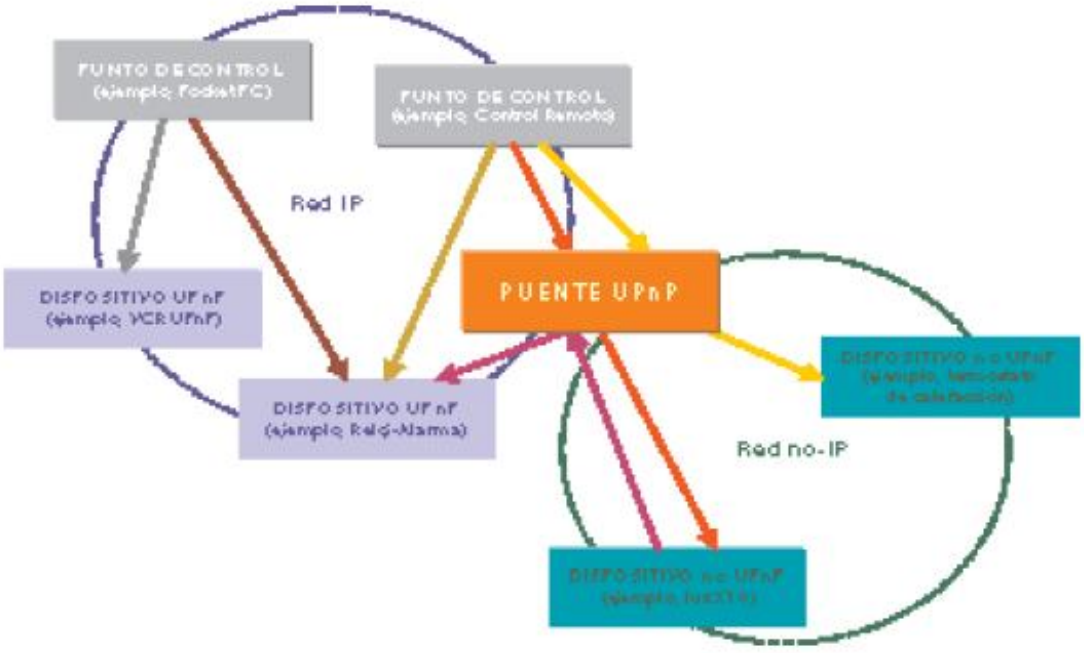


Figura 4. Ejemplo de red con UPnP Bridge

FUENTE: http://www.fundacionorange.es/areas/historico/pdf/4.pdf

2.2.3. Protocolos usados por UPnP

Como se comentó anteriormente, uno de los principales beneficios de UPnP es que utiliza protocolos estándares basados en Internet, comúnmente utilizados tanto en la propia red Internet como en redes de área local. Esto, aparte de asegurar el conocimiento de los mismos por la mayor parte de desarrolladores de

dispositivos, permite establecer la independencia de la arquitectura respecto del medio donde se implemente.

A continuación se detallan los protocolos en los que se fundamenta la tecnología UPnP:

2.2.3.1. TCP/IP

Familia de protocolos de Internet sobre el que se asientan el resto de protocolos utilizados por UPnP y que permiten la conectividad a nivel de red entre los dispositivos. Con su uso se asegura la anteriormente mencionada interoperabilidad en diferentes medios físicos e independencia de las plataformas.

Dentro de la pila de protocolos TCP/IP, se hace uso de los más comunes. Ejemplo de estos son: IP (*Internet Protocol*) como protocolo de red, TCP (*Transmission Control Protocol*) como protocolo fiable de transporte, o UDP (*User Datagram Protocol*) como protocolo no fiable de transporte.

2.2.3.2. HTTP, HTTPU, HTTPMU

HTTP supone la auténtica base sobre la cual se desarrolla toda la tecnología UPnP. No en vano, permite la comunicación al nivel de aplicación entre los dispositivos que conforman la red. HTTPU y HTTPMU son variantes de HTTP utilizados para entregar mensajes a través de UDP/IP. En el primer caso de un dispositivo a otro (*unicast*), y en el segundo, de un dispositivo al resto que están disponibles en la red UPnP (*multicast*).

2.2.3.3. SSDP (Simple Service Discovery Protocol)

Como su propio nombre indica, define cómo unos servicios de red pueden ser descubiertos por otras entidades pertenecientes a la misma red utilizando direccionamiento *multicast*. Define cómo puede un Punto de Control localizar recursos, dispositivos, de interés en la red, y cómo puede un dispositivo anunciar su presencia en la misma. UPnP lo utiliza para la fase de “Descubrimiento” de dispositivos.

Toma como protocolos base HTTPMU y HTTPU. El primero es utilizado para el envío de mensajes de búsqueda de dispositivos (utilizado por el Punto de Control) o anuncios de descubrimiento (utilizado por los dispositivos). El segundo se usa para el envío de los correspondientes mensajes de respuesta de los dispositivos a los mensajes de búsqueda.

2.2.3.4. GENA (General Event Notification Architecture)

Provee una arquitectura para el envío y recepción de mensajes de notificación utilizando HTTP sobre TCP/IP y multidifusión sobre UDP (HTTPMU). A su vez, define los conceptos de subscriptores y editores de notificaciones.

En UPnP, los formatos GENA son usado para los anuncios de presencia enviados a través de SSDP comentados anteriormente, así como para proveer la capacidad de informar de los cambios en los estados de los servicios a través de eventos. Por tanto, UPnP la utiliza para la fase de “Envío y recepción de eventos”, y en la fase de “Descubrimiento”.

2.2.3.5. SOAP (Simple Object Access Protocol)

Define el uso de XML y HTTP para ejecutar llamadas de procedimiento remoto en entornos descentralizados y distribuidos. Se ha convertido en el estándar para la comunicación basada en la tecnología RPC (*Remote Procedure Call*) en Internet. Puede hacer uso de protocolos de seguridad como SSL (*Secure Socket Layer*) para comunicaciones más seguras.

UPnP utiliza este protocolo en la fase de “Control” de los dispositivos.

SOAP se compone de cuatro partes:

En el sobre SOAP de un esquema XML se define un marco para describir lo que está en un mensaje, la forma de procesarlo, y si esta transformación es opcional u obligatoria.

Las reglas de codificación SOAP son otro esquema XML que define un conjunto de reglas para expresar instancias de tipos de datos definidos por la aplicación.

El enlace SOAP da un convenio para el uso de distintos protocolos de transporte. SOAP puede potencialmente ser utilizado en combinación con una variedad de otros protocolos de transporte (sin embargo los mensajes SOAP son las más comúnmente realizadas por HTTP).
La representación SOAP RPC. Una convención para representar llamadas a procedimientos remotos y respuestas.

El mensaje SOAP es la unidad básica de comunicación entre pares. Los mensajes SOAP están escritos en XML, SOAP es una plataforma independiente: cualquier sistema capaz de crear y analizar documentos XML pueden enviar y recibir mensajes SOAP. El poder de XML permite que los mensajes SOAP sean

bastantes complejos en la estructura y la transmisión de los tipos de datos de alta complejidad.

2.2.3.6. XML (Extensible Markup Language)

Es el formato universal para datos estructurados en la Web. Dicho de otra forma, es una forma de colocar cualquier tipo de datos estructurados, como los presentados en páginas Web, en un fichero de texto. El uso de XML como un lenguaje esquema, está definido por W3C.

UPnP lo utiliza para las descripciones de los dispositivos y servicios, y para los mensajes de control y eventos.

2.2.4. Fases de la comunicación UPnP

Como se ha comentado anteriormente, la tecnología UPnP proporciona la capacidad de comunicación entre Puntos de Control y Dispositivos. Para ello, y como consecuencia del uso de protocolos basados en Internet, define un conjunto de servidores HTTP para poder manejar las distintas fases de la comunicación.

Son seis las fases o mecanismos posibles que pueden darse durante la comunicación, Direccionamiento, Descubrimiento, Descripción, Control, Eventos y Presentación.

Figura 5. Esquema de las fases de la comunicación UPnP

FUENTE: AUTORES, Basados en UPnP Design by Example—A Software Developer's Guide to Universal Plug and Play

2.2.4.1. Direccionamiento

Más que una fase de la comunicación UPnP, el Direccionamiento, se considera el paso previo al desarrollo de la misma. Se trata de que los dispositivos que integren la red UPnP obtengan una dirección IP con la que se identifiquen y puedan comunicarse con el resto de miembros. Para ello, el dispositivo debe disponer de un cliente Dynamic Host Configuration Protocol (DHCP) y buscar un servidor DHCP disponible en la red que le asigne una IP. De no encontrar ningún servidor, debe utilizar la función de Auto-IP, con la que obtiene una IP por él mismo.

2.2.4.2. Descubrimiento

Una vez que el dispositivo tiene asignada una IP única para conectarse a la red, está en condiciones de pasar a la primera de las fases de la comunicación UPnP, el Descubrimiento.

Para esta fase se hace uso del mencionado protocolo SSDP. Mediante este protocolo el dispositivo se agrega a la red. Este proceso le permite anunciar sus servicios a los Puntos de Control u otros dispositivos. A su vez, a un Punto de Control, le permite la búsqueda de dispositivos en la red.

Los mensajes enviados por el dispositivo, tanto de descubrimiento como los de respuesta, contienen información sobre algunos aspectos esenciales e importantes de los dispositivos y servicios que anuncia. Cabe destacar, la localización o dirección en formato *Uniform Resource Locator* (URL), del documento XML de descripción del dispositivo o descriptor. Esa dirección proporciona a los puntos de control la información que estos necesitan recuperar del dispositivo y las descripciones de los servicios. Los documentos de descripción son recuperados por los puntos de control y se analizan para entender

todo acerca de ese dispositivo. Los vendedores pueden añadir extensiones más allá de las funciones básicas e incluyen las extensiones de los documentos de descripción. Ese mecanismo de extensión permite a un punto de control preferir una interfaz opcional o específica del fabricante sobre el estándar.

Todos los servicios dentro de un dispositivo se cargan en tres direcciones URL que proporcionan la información necesaria para que los puntos de control puedan acceder a la comunicación con dichos servicios.

El Control URL, es donde los puntos de control realizan sus solicitudes de preparacion para controlar el servicio. Los vendedores de un dispositivo UPnP la específican para cada dispositivo.

El EventSubURL, es donde los puntos de control preparados, solicitan la suscripción a los eventos.
Existe un EventSubURL para cada servicio en un dispositivo.

El DescriptionURL, dice a los puntos de control la ubicación para recuperar el documento de descripción del servicio. La descripción del servicio XML documento se devuelve a través de una solicitud HTTP GET. A veces, incluso después de reconocer un tipo de dispositivo UPnP aprobado por el Foro UPnP, un punto de control descubre documentos individuales de descripción del servicio. Dado que algunas interfaces de servicios son opcionales, no todos los vendedores podrán aplicar una capacidad particular. Por ejemplo, una impresora UPnP que no admite la impresión en color es probable que decida no implementar medidas para ajustar las propiedades del color.

La figura 6 ilustra los pasos adoptados por una aplicación de punto de control UPnP para descubrir las capacidades de un dispositivo y la figura 7 muestra el

intercambio de mensajes entre puntos de control y dispositivos ocurridos durante el descubrimiento.

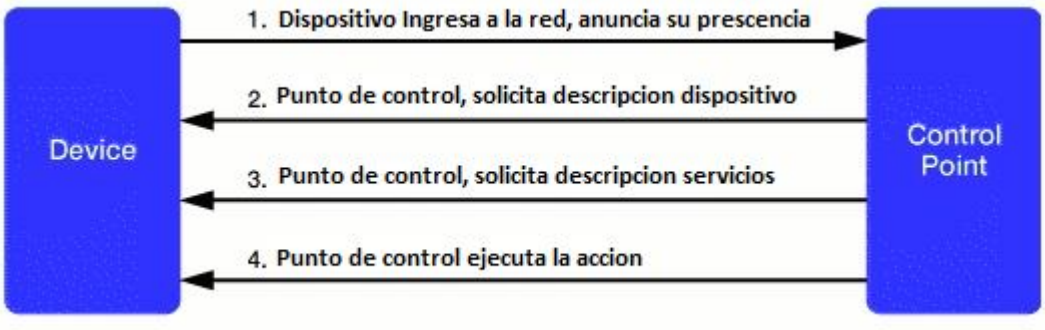


Figura 6. Recuperación de la descripción de dispositivos y servicios

FUENTE: AUTORES basados en http://www.artima.com/spontaneous/upnp_digihome.html

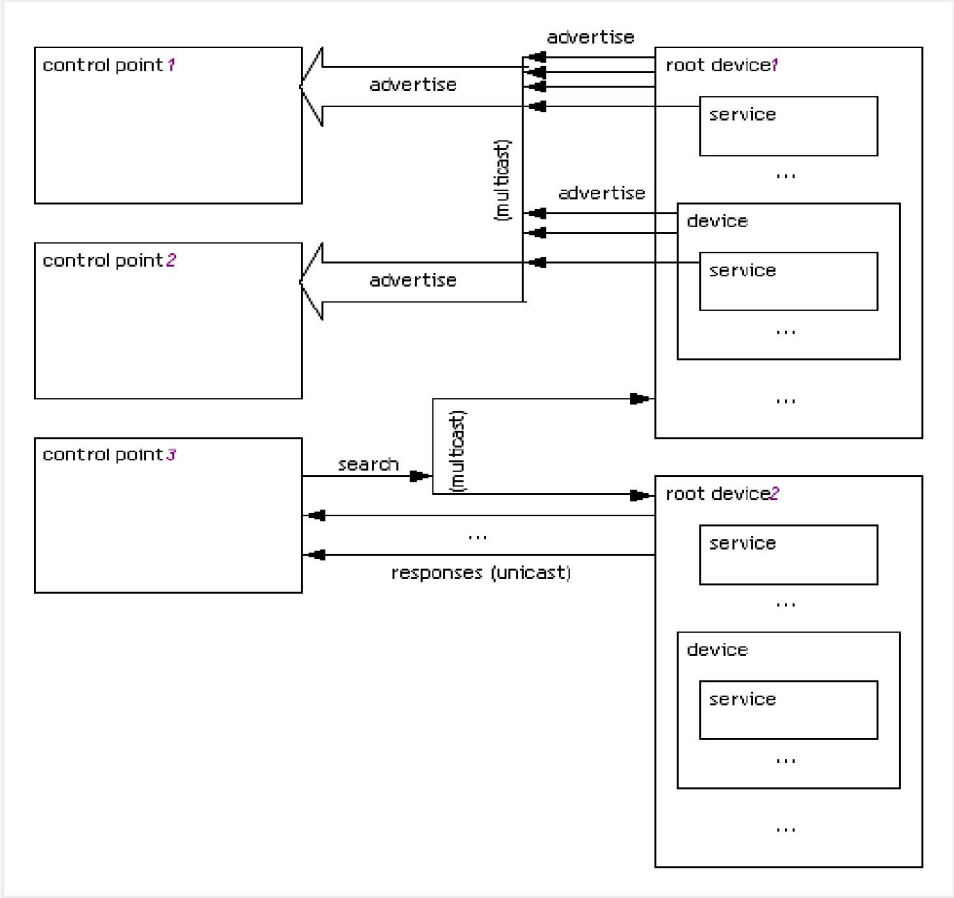


Figura 7. Intercambio de mensajes en la fase de Descubrimiento UPnP.

FUENTE: http://www.upnp.org/specs/arch/UPnP-arch-DeviceArchitecture-v1.1.pdf

2.2.4.3. Descripción

Después de que el Punto de Control ha descubierto al dispositivo, puede conocer las capacidades de este, sus servicios y sus dispositivos embebidos, así como también puede interactuar con él, u obtener su descripción. Para ello debe recuperar el documento XML con esta descripción, a partir de la URL que, como se comentó en el apartado anterior, se especifica en el mensaje de descubrimiento.

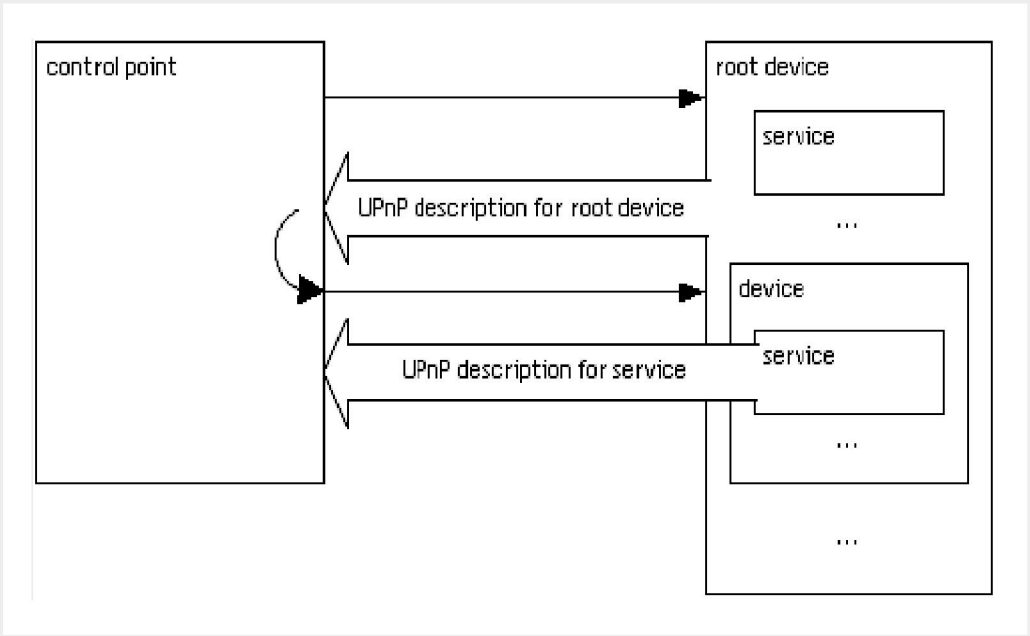


Figura 8. Esquema de la fase de Descripción UPnP

FUENTE: http://www.upnp.org/specs/arch/UPnP-arch-DeviceArchitecture-v1.1.pdf

La descripción completa del dispositivo está dividida en dos partes lógicas, la descripción del dispositivo como fabricante, marca, modelo y número de serie y las descripciones de los correspondientes servicios asociados a éste. Un documento

de descripción de servicio contiene información detallada sobre el servicio de un dispositivo, las acciones que el servicio proporciona, y los parámetros del servicio y valor de retorno.

La Descripción permite a los dispositivos presentar la lista de funcionalidades que ofrecen. Las descripciones de dispositivos y sus servicios están contenidos en los documentos de descripción basada en XML.

En la figura numero 9 se puede apreciar las dos categorías manejadas para la descripción de los dispositivos, del lado izquierdo la información del fabricante y del lado derecho los servicios y propiedades incluidos en el dispositivo.

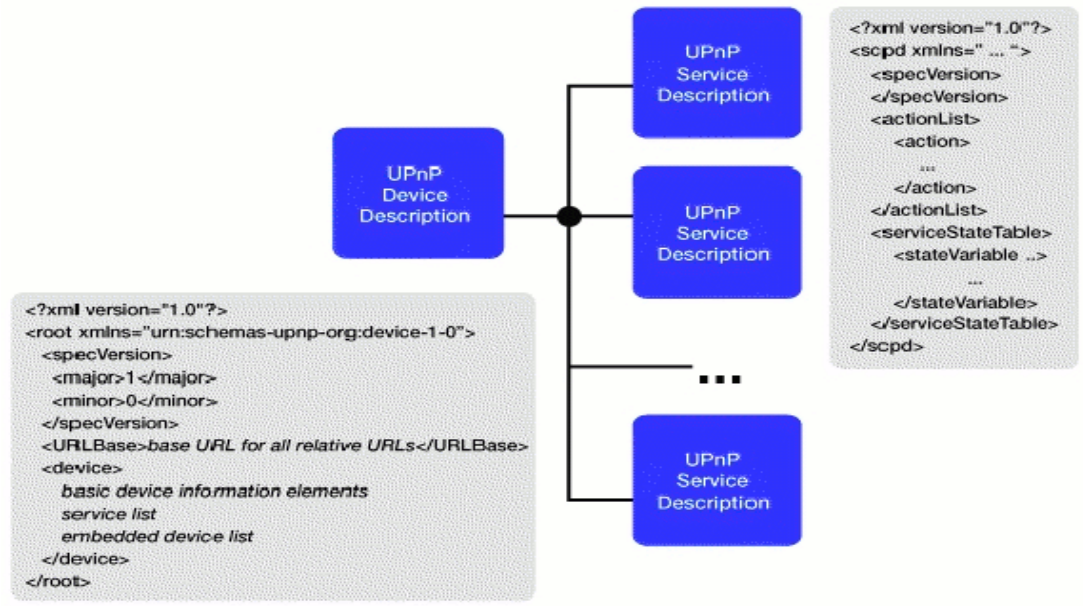


Figura 9. Ilustración del dispositivo y la jerarquía de descripción del servicio.

FUENTE: Autores basados en http://www.artima.com/spontaneous/upnp_digihome.html

Documento de descripción del dispositivo (DCPD)

Este documento contiene información acerca del fabricante del dispositivo, información de fabricación, como nombre y tipo de dispositivo, numero de serie,

Urls del fabricante, etc. En el mismo documento se listan los distintos tipos de servicios que implementa el dispositivo. Para cada uno se incluye datos tales como, el nombre, una URL para su descripción, una URL para el mecanismo de Control, y una URL para el mecanismo de Eventos. Además, incluye una descripción de todos los dispositivos embebidos que contiene el dispositivo raíz, y una URL de presentación del dispositivo.

En el siguiente cuadro de texto se muestra el documento XML tipo para la descripción de un dispositivo:

```
<?xml version="1.0" encoding="utf-8"?>
<root xmlns="urn:schemas-UPnP-org:device-1-0">

<specVersion>

<major>1</major>

<minor>0</minor>

</specVersion>

<URLBase>URL base para todas las urls relativas</URLBase>

<device>

<deviceType>urn:schemas-UPnP-org:device:TipoDeDispositivo:version</deviceType>

<friendlyName>nombre corto</friendlyName>
<manufacturer>nombre del fabricante</manufacturer>
<manufacturerURL>url del fabricante</manufacturerURL>
<modelDescription>nombre largo y descriptivo del dispositivo</modelDescription>
<modelName>nombre del modelo</modelName>
<modelNumber>numero del modelo</modelNumber>
<modelURL>url al sitio del modelo</modelURL>
<serialNumber>numero de serie del fabricante</serialNumber>
<UDN>identificador unico y universal del dispositivo</UDN>
<UPC>codigo universal del producto</UPC>

<iconList>
<icon>
<mimetype>formato imagen</mimetype>
<width>pixeles horizontales</width>
<height>pixeles verticales</height>
<depth>profundidad del color</depth>
```

```
<url>URL de localizacion del icono</url>
</icon>
...
</iconlist>
<serviceList>
<service>
<serviceType>urn:schemas-UPnP-org:service:TipoServicio:version</serviceType>
<serviceId>urn:UPnP-org:serviceId:IDServicio</serviceId>
<SCPDURL>URL para la descripcion del servicio</SCPDURL>
<controlURL>URL para control</controlURL>
<eventSubURL>URL para eventos</eventSubURL>
</service>

... otros servicios definidos por el Foro UPnP
... otros servicios de valor añadido creados por el fabricante
</serviceList>
<deviceList>
... otros dispositivos embebidos definidos por el Foro UPnP
... otros dispositivos embebidos añadidos por el fabricante
</deviceList>
</device>
</root>
```

FUENTE: Autores basados en la especificación del fórum UPnP

Documento descriptor del servicio (SCPD)

En ese documento se listan las variables de estado que definen el estado del servicio, así como el conjunto de acciones, con sus respectivos argumentos de entrada o salida, que están disponibles para ser invocados por el controlador, el Punto de Control.

A continuación se muestra el documento XML tipo para la descripción de un servicio:

```


</action>
... otras acciones definidas por el Foro UPnP
... otras acciones definidas por el fabricante
</actionList>
</scpd>
```

```
<step>incremento</step>
</allowedValueRange>
</stateVariable>
... otras variables de estado definidas por el Foro UPnP
... otras variables de estado definidas por el fabricante
</serviceStateTable>
<actionList>
<action>
<name>nombre accion</name>
<argumentList>
<argument>
<name>nombre parametro</name>
<direction>direccion: entrada (in) o salida (out)</direction>
<relatedStateVariable>variable de estado relacionada</relatedStateVariable>
</argument>
... otros argumentos definidos por el Foro UPnP

<?xml version="1.0" encoding="utf-8"?>
<scpd xmlns="urn:schemas-UPnP-org:service-1-0>
<specVersion>
<major>1</major>
<minor>0</minor>
</specVersion>
<serviceStateTable>
<stateVariable sendEvents="yes"> (yes: si se envía evento cuando cambia)
<name>nombre variable</name>
<dataType>tipo de datos</dataType>
<allowedValueList>
<allowedValue>valor enumerado</allowedValue>
... otros valores definidos por el Foro UPnP
</allowedValueList>
</stateVariable>
<stateVariable sendEvents="no">
<name>nombre variable</name>
<dataType>tipo de datos</dataType>
<defaultValue>valor por defecto</defaultValue>
<allowedValueRange>
<minimum>valor minimo permitido</minimum>
<maximum>valor maximo permitido</maximum>


</action>
... otras acciones definidas por el Foro UPnP
... otras acciones definidas por el fabricante
</actionList>
</scpd>

<step>incremento</step>
</allowedValueRange>
```

FUENTE: Autores basados en la especificación del fórum UPnP

Como se mencionó anteriormente y a la vista de los cuadros de texto, se tratan de documentos XML, escritos por el fabricante del dispositivo, con una estructura estándar definida por el Foro UPnP.

Estos descriptores, tanto de dispositivos como de los servicios, son recuperados por el Punto de Control mediante solicitudes HTTP GET.

Tras esta fase, el Punto de Control ya dispone de la información necesaria para poder proceder con cualquiera de los subsiguientes pasos, Control, Eventos, o Presentación. Siendo es el motivo por el que en la figura 5, estas etapas aparezcan al mismo nivel y justo por encima de la fase de Descripción.

2.2.4.4. Control

Una vez obtenido el conocimiento suficiente sobre el dispositivo y sus servicios, un Punto de Control puede invocar acciones definidas por los servicios. Para ello, envía mensajes de control a la URL de control del servicio correspondiente (proporcionada en la descripción del dispositivo). En respuesta, el servicio genera un mensaje en el que se incluye la respuesta a la invocación de la acción, si procede, o códigos de error.

Los mensajes de control también se codifican en XML utilizando el formato que define el protocolo SOAP descrito en el apartado “Protocolos usados por UPnP”.

UPnP se basa en la Simple Object Access Protocol (SOAP) para el control del dispositivo. SOAP reúne a XML y HTTP para proporcionar una solución de mensajería basada en web y el mecanismo de llamada a procedimiento remoto.

XML expresa el contenido del mensaje, mientras que HTTP envía mensajes a su destino. SOAP se especifica como un conjunto de convenciones que gobiernan el formato y las reglas de procesamiento de mensajes SOAP.

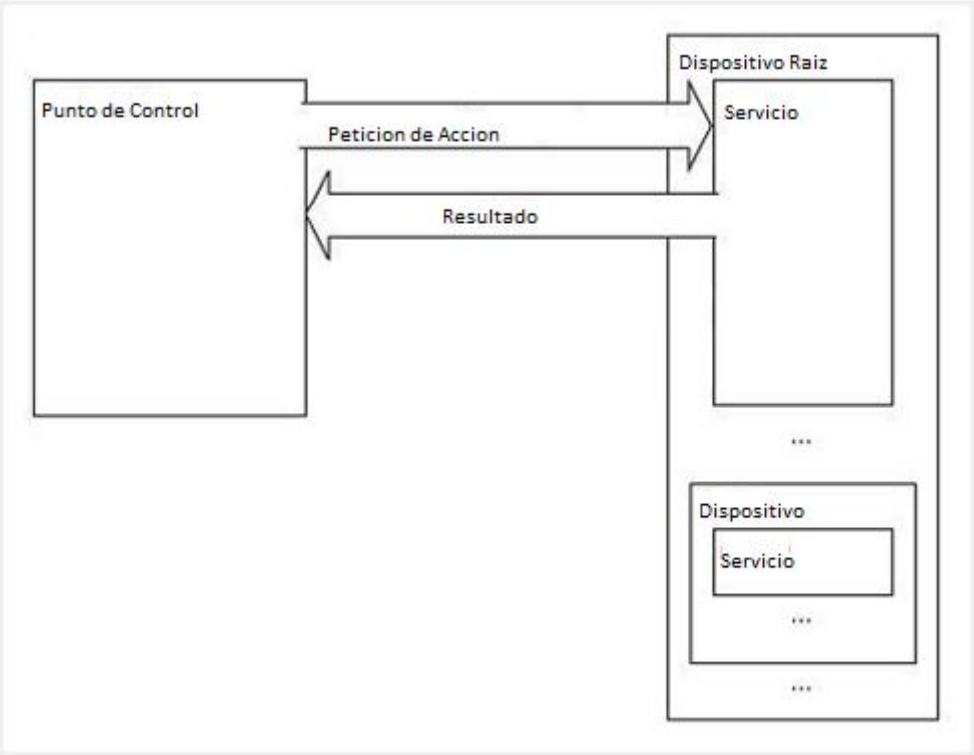


Figura 10. Esquema del mecanismo de control UPnP

FUENTE: Autores basados en http://www.upnp.org/specs/arch/UPnP-arch-DeviceArchitecture-v1.1.pdf

2.2.4.5. Eventos

Al igual que en el mecanismo anterior, tras disponer el Punto de Control de la información que le facilitan las descripciones del dispositivo y sus servicios, ya está en condiciones de proceder con el mecanismo de eventos.

En la etapa de Descripción se dijo que la descripción de un servicio incluye una lista de variables de estado. Algunas de estas variables sufren modificaciones por cambios durante el desarrollo del programa del propio servicio, o bien como consecuencia de la invocación de alguna de las acciones de control definidas para éste servicio.

Si una o más de estas variables generan eventos cuando son modificadas, entonces, el dispositivo debe publicar dichas actualizaciones. Para que un Punto de Control pueda escuchar estos eventos debe haberse subscrito previamente utilizando la URL que se le facilita para tal efecto en la descripción del servicio, también se suscribe utilizando un tiempo de suscripción para la notificación de eventos. Una suscripción es una solicitud para recibir todos los cambios de variable de estado. UPnP no provee un mecanismo para suscribirse a los eventos de forma variable por variable. Si un servicio acepta la solicitud de suscripción, el servicio responde con un identificador único de suscripción y la duración de la suscripción. El identificador único permite que el punto de control pueda hacer referencia a la suscripción de las solicitudes posteriores al servicio, tales como la renovación o cancelación de la suscripción. La duración especifica la longitud de tiempo que la suscripción es válida y gestionada por el servicio. Si una o más de las variables de estado de un servicio son evented(eventos generados), el servicio publica actualizaciones a los suscriptores cuando se presente el cambio de estas variables.

De esta forma, el Punto de Control, estaría enterado de cualquier modificación permitiéndole generar un modelo del estado de este servicio en tiempo de ejecución.

Los mensajes de eventos, enviados por los servicios a sus subscriptores, contienen de una a más variables de estado y su valor actual. También estos

mensajes se expresan en XML, y se formatean utilizando la arquitectura GENA, y TCP para su transporte.

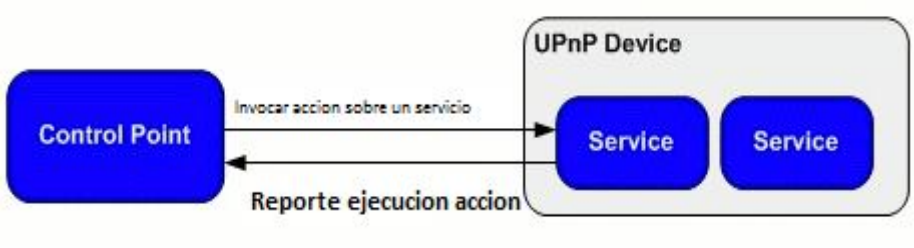


Figura 11. Ilustración de notificación de eventos UPnP

FUENTE: AUTORES basados en http://www.artima.com/spontaneous/upnp_digihome.html

Cuando una suscripción expira, el identificador de suscripción deja de ser válido, y en el editor se detiene el envío de mensajes de eventos para el suscriptor. Si el abonado intenta enviar cualquier mensaje que no sea un mensaje de solicitud de suscripción, una solicitud de renovación de suscripción o un mensaje de cancelación de suscripción, el editor rechaza el mensaje debido a que la suscripción de identificador no es válida.

Todas las suscripciones deben renovarse periódicamente para que los puntos de control puedan seguir recibiendo notificaciones. Para mantener una suscripción activa, un punto de control debe enviar un mensaje de renovación antes de que venza la suscripción. El mensaje de renovación se envía a la misma URL que el mensaje de suscripción original, pero el mensaje de renovación no incluye una dirección URL de entrega de mensajes de sucesos. En cambio, el mensaje de renovación incluye el identificador de suscripción recibidas en el mensaje inicial de aceptación de la suscripción. Cuando un punto de control ya no quiere recibir eventos de un servicio, el punto de control puede cancelar su suscripción.

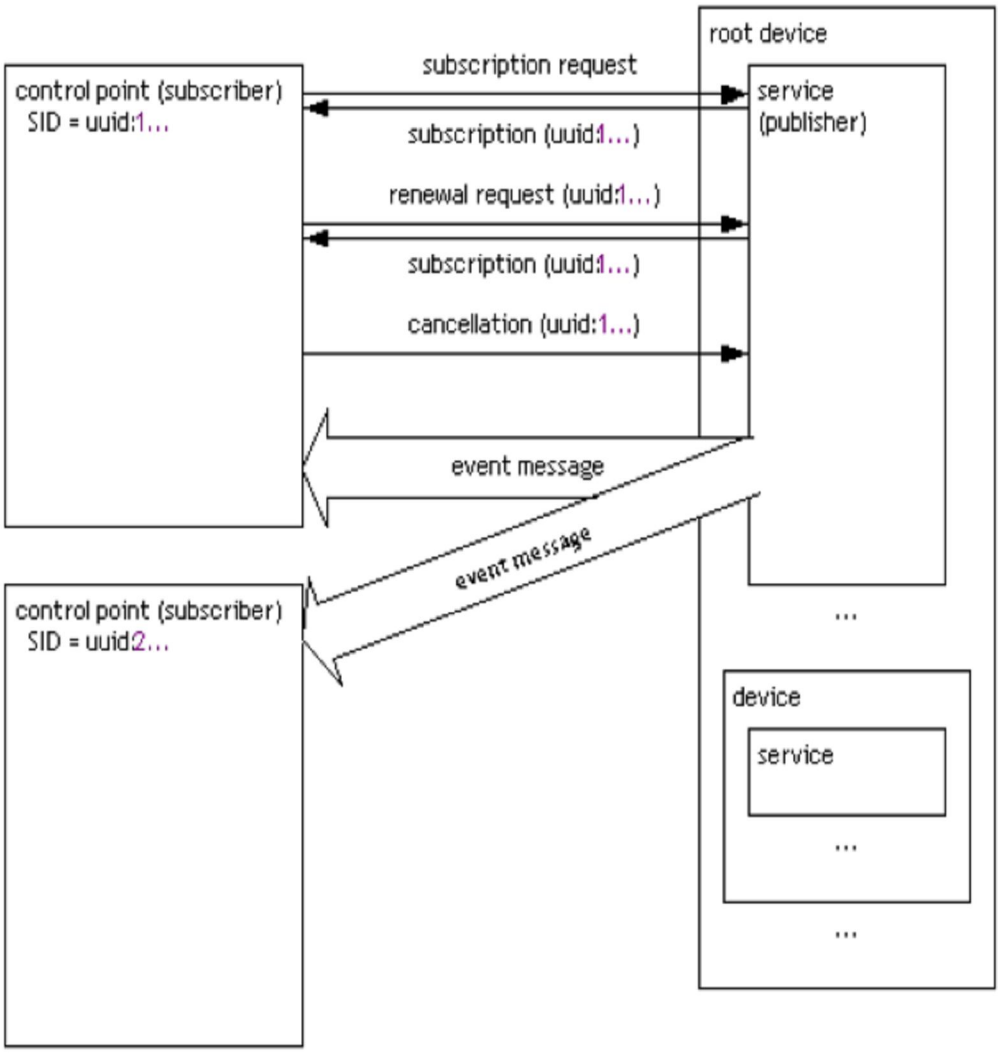


Figura 12. Esquema del mecanismo de notificación de eventos UPnP

FUENTE: http://www.upnp.org/specs/arch/UPnP-arch-DeviceArchitecture-v1.1.pdf

2.2.4.6. Presentación

Alcanzado el mismo nivel de conocimiento de las capacidades del dispositivo que en los dos mecanismos anteriores, el Punto de Control puede recuperar una página en formato *HyperText Markup Language* (HTML) a partir de la URL de presentación que se incluye en la descripción del servicio correspondiente. Esta

página es obtenida mediante una solicitud HTTP GET. Se carga en un explorador web y, en función de las capacidades de la misma, permite al usuario controlar al dispositivo y/o visualizar información acerca de su estado actual.

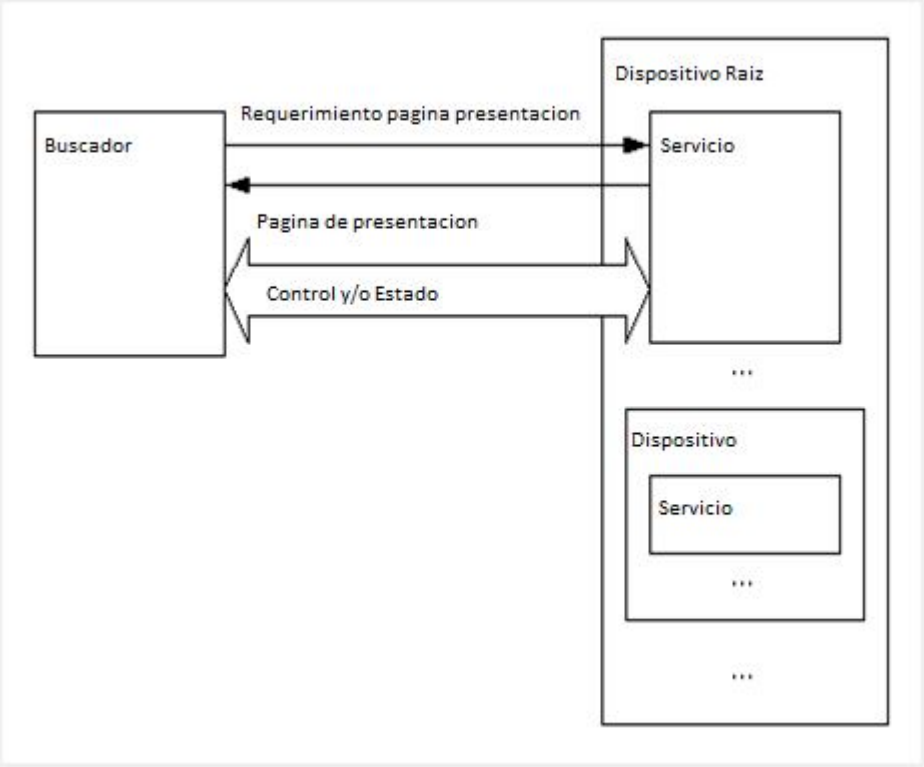


Figura 13. Esquema del mecanismo de Presentación UPnP

FUENTE: Autores basados en http://www.upnp.org/specs/arch/UPnP-arch-DeviceArchitecture-v1.1.pdf

2.2.5. Arquitectura UPnP para Dispositivos

UPnP se basa en formatos y protocolos existentes, incluyendo HTTP, XML, SOAP, el Modelo de Objetos de Documento, y multicast IP. Estos elementos se utilizan conjuntamente para definir un marco que permita la definición, el descubrimiento y control de dispositivos de red.

Comités o grupos de trabajo, del Foro UPnP formaron la base del marco común de colaboración para definir los tipos estándar de dispositivos. En el documento de la arquitectura del dispositivo UPnP se detallan los protocolos y convenios necesarios de los dispositivos UPnP, y explica el patrón básico que todos los dispositivos UPnP siguen en su funcionamiento:

Direccionamiento: El dispositivo se une a la red, adquiriendo una dirección única que las entidades pueden utilizar para comunicarse con el dispositivo.

Descripción: El dispositivo se resume sus servicios y prestaciones en un formato estándar.

Descubrimiento: El dispositivo se encuentra en los puntos de control que aprender acerca de las capacidades del dispositivo mediante la recuperación de una descripción del dispositivo.

Control: El dispositivo se ocupa de las peticiones de los puntos de control.

Concurso Completo: El dispositivo notifica a los puntos de control registrados los cambios de estado interno.

Presentación: El dispositivo proporciona una interfaz de administración basada en HTML para permitir la manipulación directa y el control del dispositivo.

Cada fase UPnP se relaciona con protocolos de red. El protocolo TCP / IP y HTTP proporciona la conectividad de red básica para UPnP. La figura 14 muestra el conjunto de protocolos básicos de UPnP en forma de pila.

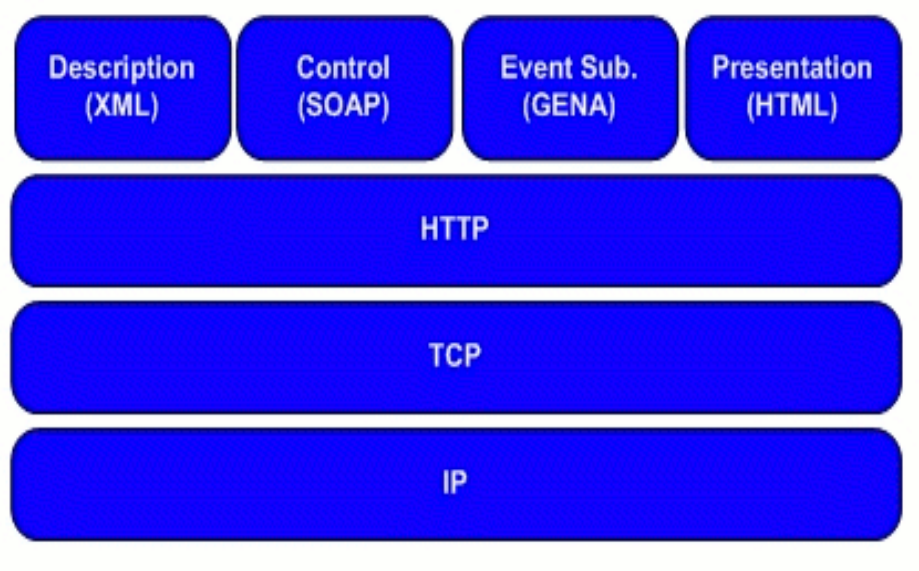


Figura 14. Pila de Protocolos usados por UPnP

FUENTE: http://www.artima.com/spontaneous/upnp_digihome.html

Algunos aspectos de UPnP exigen al emisor comunicarse con varios destinatarios. Por ejemplo, el dispositivo UPnP posee un mecanismo de descubrimiento que permite a los dispositivos enviar anuncios a todos sobre la presencia de los otros dispositivos en la red, haciéndoles saber la disponibilidad del dispositivo. Para ello, UPnP se basa en una forma de HTTP que se envían a través de multidifusión UDP, llamado HTTPMU. En algunos casos, los mensajes de respuesta se envían directamente a la fuente usando HTTP a través de UDP unicast, haciendo un llamado al HTTPU. El protocolo de multidifusión y su posición en la pila se muestra en la Figura 15.

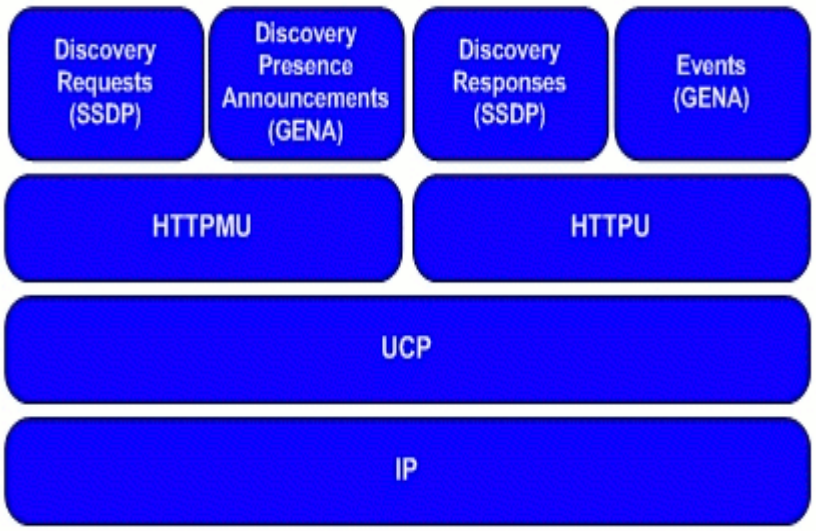


Figura 15. Pila de protocolo HTTPMU y HTTPU

FUENTE: http://www.artima.com/spontaneous/upnp_digihome.html

2.2.5.1. Dispositivos

Un dispositivo UPnP puede ser pensado como un contenedor lógico con un tipo único. Cada dispositivo UPnP puede ofrecer cualquier número de servicios, cada servicio con su propio tipo de servicio único. Por sí mismo, un dispositivo no hace más que proporcionar la auto-descripcion de la información del fabricante, nombre del modelo y número de serie. los servicios de un dispositivo y proporcionar su funcionalidad real.

Un dispositivo UPnP puede contener otros dispositivos. Que según la composición lógica de dispositivos ofrece la flexibilidad de diseño para determinar la forma de aplicar el dispositivo. También permite que el dispositivo integrado pueda descubrir y utilizar de forma independiente del dispositivo contenedor principal

otros servicios y dispositivos disponibles. La Figura 16 muestra un detallado diagrama UML de clases para un dispositivo UPnP.

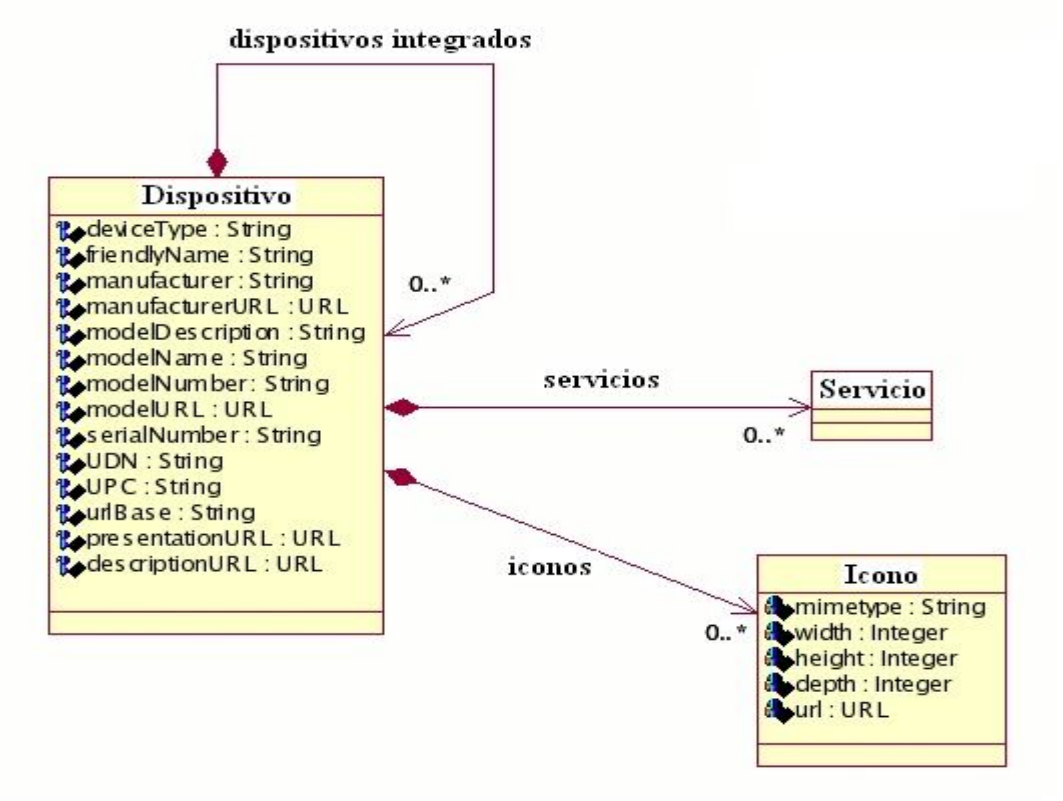


Figura 16. Diagrama de clase: Dispositivos UPnP

FUENTE: Autores basados en http://www.artima.com/spontaneous/upnp_digihome.html

2.2.5.2. Servicios

Cada servicio en un dispositivo UPnP puede contener cualquier número de acciones, cada acción tiene un conjunto de parámetros de entrada y un valor de retorno opcional. Cada acción tiene un nombre y un conjunto opcional de argumentos. Cada argumento tiene un nombre, un valor, y una dirección. La dirección puede ser de entrada o salida (pero no ambos), dependiendo de si el

argumento es pasado a la acción o se devuelve desde la acción a la persona que llama. Cada acción también puede tener un valor de retorno que proporciona el resultado de la acción.

Aquellos que estén familiarizados con los sistemas de modelo orientado a la interfaz de objetos, como Microsoft COM o CORBA, se pueden trazar una analogía: un dispositivo UPnP es similar a un objeto, los servicios son como las interfaces admitidos por el objeto, y las acciones en un servicio son como las funciones en una interfaz.

Cada servicio UPnP también tiene un servicio de tipo Uniform Resource Identifier (URI) que identifica de forma exclusiva el servicio. La Norma de tipos de servicio se define por un comité de trabajo del Foro UPnP. Cada servicio también tiene una serviceId URI que identifica el servicio entre todos los servicios de un dispositivo. No hay dos servicios que puedan compartir la misma serviceId. Los servicios que un dispositivo debe implementar son determinados por el tipo del dispositivo. Las comisiones de trabajo del Foro UPnP normalizan el conjunto de servicios que un tipo de dispositivo debe admitir.

Cada servicio también podrá mantener las variables que representan el servicio de estado actual. Un servicio puede tener cero o más variables. Cada variable de estado tiene un nombre, un tipo, un valor predeterminado, y un valor actual. También cuenta con un conjunto de valores permitidos para describir la gama de valores de variables permitidas. Cualquier variable de estado puede desencadenar eventos en los cambios de estado, cuando se producen estos eventos está determinada la ejecución de un servicio.

Si una variable dispara un evento para indicar un estado, esa variable se dice que es evented. Cada argumento de entrada a una acción se asocia con una de las variables de estado del servicio, denominada variable de argumento del estado relacionado. Uno de los argumentos de una acción puede ser designado como

valor de retorno de la acción. El valor de retorno proporciona el resultado de la acción a la persona que llama. La figura 17 se muestra un diagrama de clases UML para un servicio UPnP.

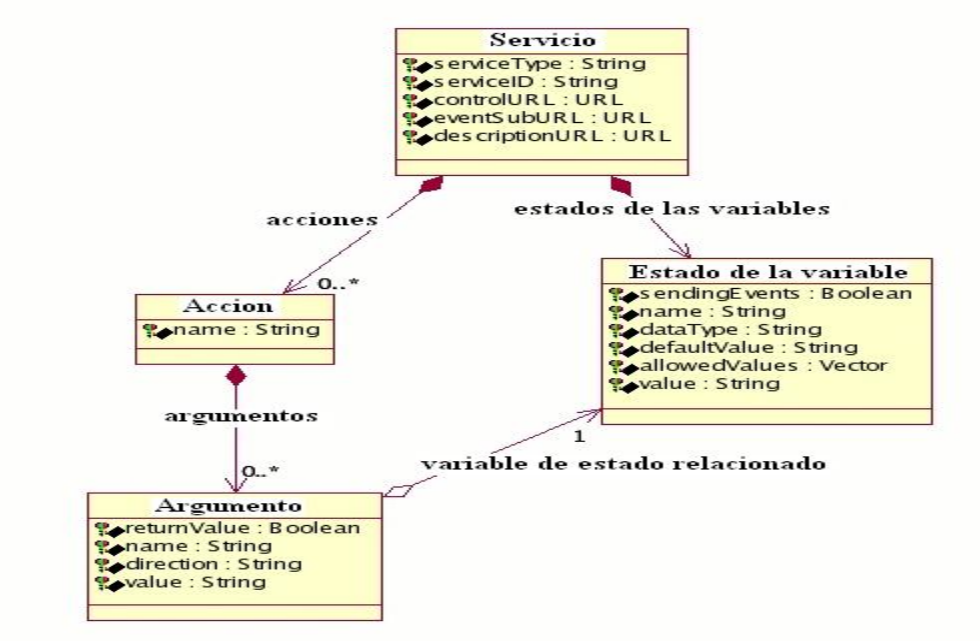


Figura 17. Diagrama de Clase: Servicios UPnP

FUENTE: Autores basados en http://www.artima.com/spontaneous/upnp_digihome.html

2.2.5.3. Punto de control

Un punto de control es una entidad de red que invoca la funcionalidad de un dispositivo. En términos informáticos de cliente/ servidor, el punto de control es el cliente y el dispositivo es el servidor.

Los puntos de control ejecutan acciones sobre los servicios y proporcionan parámetros de entrada necesarios y reciben cualquier parámetro de salida y, posiblemente, un valor de retorno. La Figura 18 muestra un punto de control que

invoca una acción en un dispositivo UPnP que implementa un solo tipo de dispositivo UPnP que contiene dos servicios.

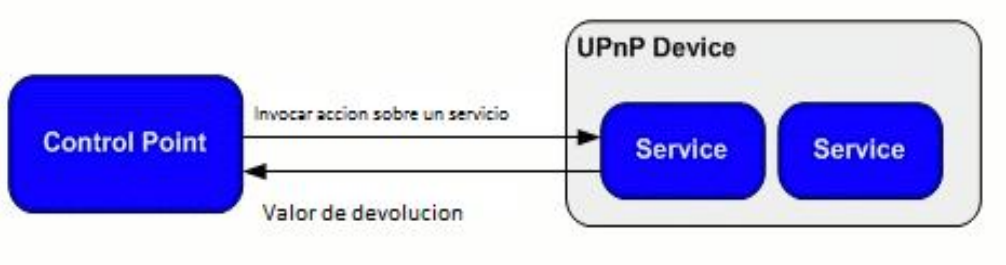


Figura 18. Invocación de la acción de un servicio por un punto de control UPnP

FUENTE: Autores basados en http://www.artima.com/spontaneous/upnp_digihome.html

Como ilustra el gráfico, un punto de control descubre los dispositivos, invoca las acciones sobre los servicios de un dispositivo, y se adhiere a las notificaciones de eventos. Un dispositivo, por el contrario, responde a las acciones invocadas, envía eventos cuando el estado de las variables cambia, y se apoya en una página Web para el control administrativo (una página de presentación en UPnP).

2.2.6. El API (Interfaz de Programación de la Aplicación)

Dos componentes esenciales de un sistema basado en UPnP son el punto de control y el dispositivo. Existen API´s para los dos. El modelo arquitectónico se describe en la figura 19. Hay 5 capas en el modelo. La capa superior es la aplicación que se construye en la parte superior de la API, la segunda capa el API es una colección de varias clases y la aplicación se construye a partir de estas clases.

Los diagramas de clases para la API del punto de control y del dispositivo se presentan en la Figura 20. La aplicación del dispositivo específico es una extensión de la clase Dispositvo; la aplicación de control se extiende de la clase

Descubrimiento específicamente. Dentro del punto de control, la clase Dispositivo y la clase Servicio representan exactamente al dispositivo y a los servicios reales.

La Clase GUIControl representa el marco para la interfaz de usuario si es necesario.

El nivel 3 de las capas incluye tres componentes que implementan los tres protocolos SSDP, SOAP y Gena. Los componentes en la capa 3 son iniciadas por la API. La línea punteada en la figura 20 representa una función de devolución de llamada, la dirección de la flecha representa la dirección de la transferencia de datos. La capa 4 representa dos protocolos conocidos: UDP y TCP. Para llevar a cabo el descubrimiento, el componente SSDP utiliza UDP.

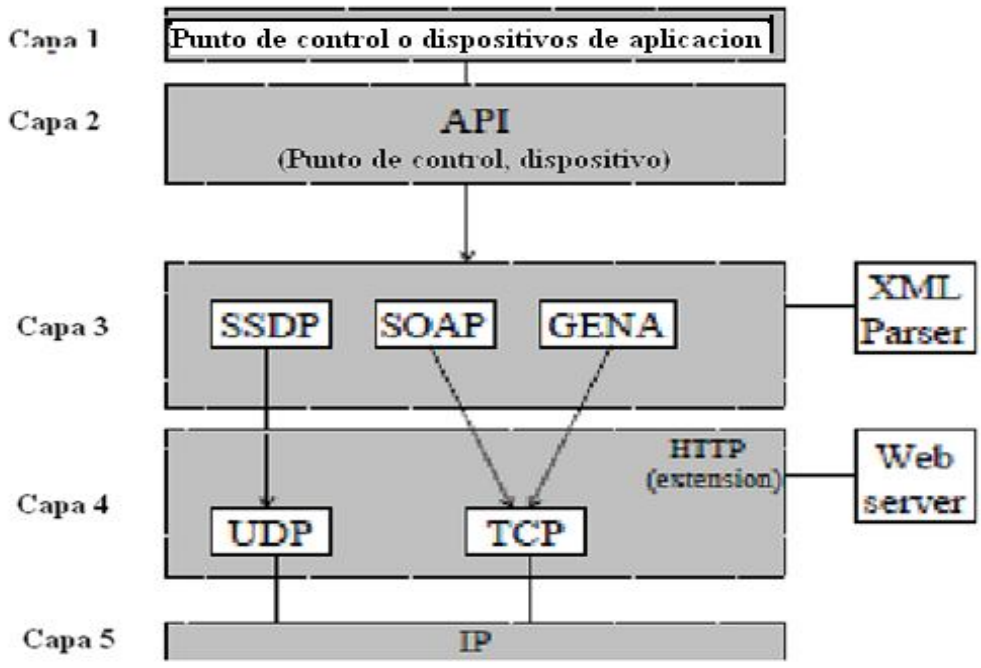


Figura 19. Modelo arquitectónico del UPnP

FUENTE: www.win.tue.nl/~johanl/projects/EES5413/UPnP_IASTED.pdf

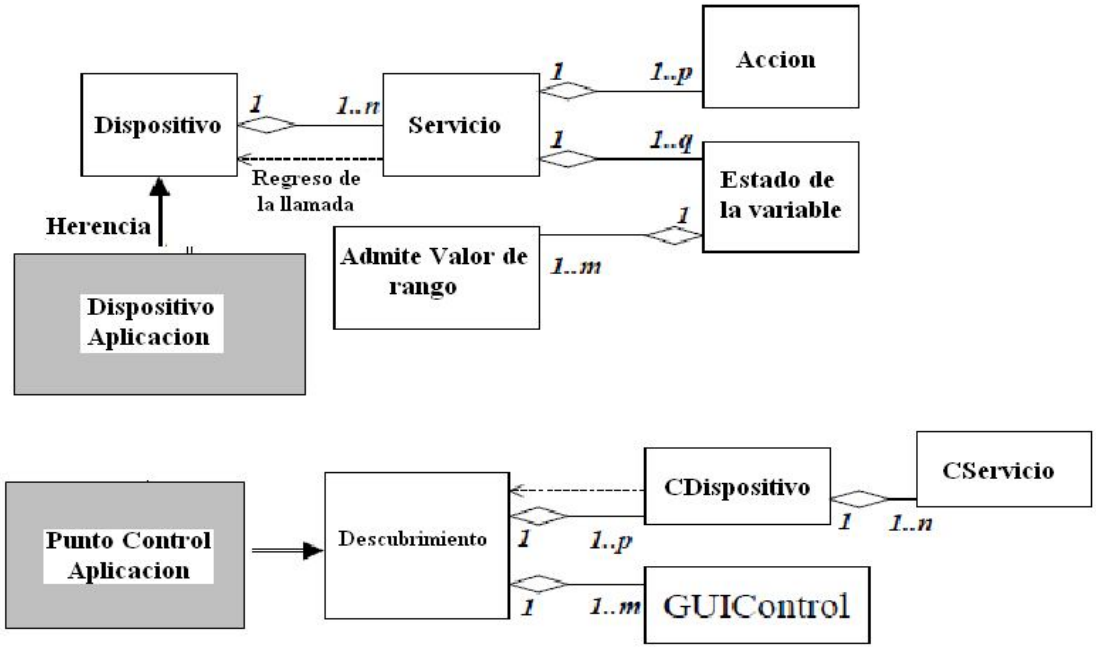


Figura 20. Diagrama de clases para el API UPnP

FUENTE: www.win.tue.nl/~johanl/projects/EES5413/UPnP_IASTED.pdf

Para el control (SOAP) y concurso completo (GENA), se utiliza TCP. El formato de transferencia de datos entre la capa 3 y capa 4, es conforme al protocolo HTTPe que es una versión ampliada de HTTP/1.1 con el propósito de cumplir con el requisito de la SSDP, GENA y SOAP. En la capa 4 dos clases son esenciales UDPListener y TCPListener. UDPListener es un servidor HTTPe ejecutándose encima de UDP para manejar el proceso de descubrimiento de servicios en el SSDP. TCPListener es un servidor HTTPe ejecutándose encima de TCP para hacer frente a las exigencias de SOAP y Gena.

Dado que IP es la última capa, cada elemento en el sistema debe ser habilitado para IP.

Hay dos componentes que no pertenecen a ninguna capa. El componente analizador CMarkup puede ser invocado por cualquiera de los componentes en la

capa 3 para analizar el contenido XML. El servidor namedWeb componente es un servidor HTTP/1.1 llevando a cabo sólo en el dispositivo. Este servidor maneja las solicitudes para el dispositivo y descripciones de los servicios.

Para explicar el sistema con más detalle, se describe brevemente cómo trabajan SSDP, SOAP y el GENA. Cuando un dispositivo se une a una red, se anuncia a la red con una cierta frecuencia. Llamamos a esta etapa anuncio. Cuando un punto de control se une a una red, busca el dispositivo que puede controlar. Llamamos a este descubrimiento. El protocolo SSDP incluye tanto Anunciamiento y Descubrimiento.

El módulo SSDP incluye las siguientes clases:

✎ AdvListener: Es un servidor HTTPe que se ejecuta en el punto de control para manejar el anunciamiento de los dispositivos.

✎ AdvSolver: Procesa los datos facilitados por AdvListener.

✎ MSearch: Inicia el descubrimiento del lado del punto de control.

✎ S_RSolver: Controla las respuestas de los dispositivos cuando el punto de control concluye los procesos de descubrimiento.

✎ D_Solver: Inicia el anuncio del lado dispositivo.

✎ SSDPSolver: Procesa los datos facilitados por UDPLis-dora.

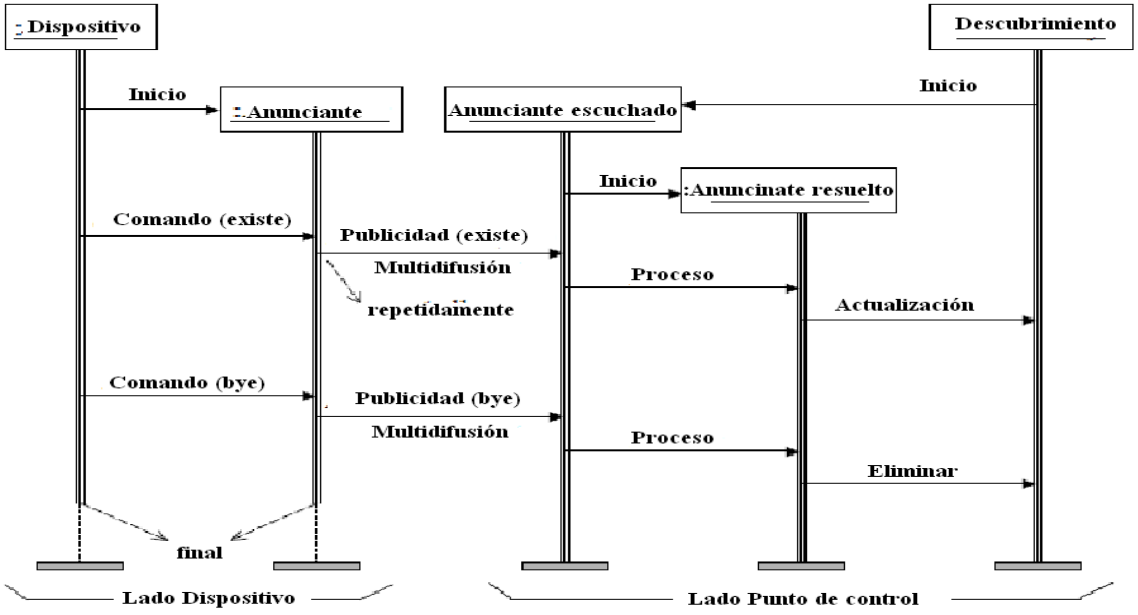


Figura 21. Descripción del proceso de Anunciamiento

FUENTE: www.win.tue.nl/~johanl/projects/EES5413/UPnP_IASTED.pdf

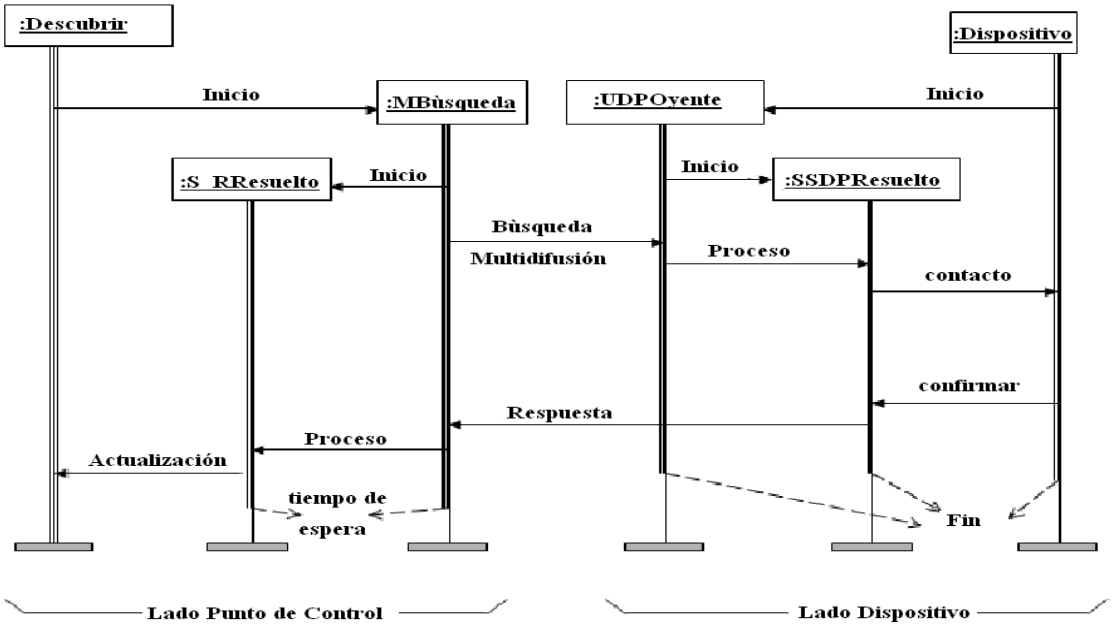


Figura 22. Descripción del proceso de Descubrimiento

FUENTE: www.win.tue.nl/~johanl/projects/EES5413/UPnP_IASTED.pdf

Los diagramas de secuencia UML para el anuncio y descubrimiento se muestran en las Figuras 21 y 22.

Los componente SOAP incluyen sólo la clase namedSOAP Solver corriendo en el dispositivo para manejar los comandos de control expedidos por el punto de control. El diagrama de secuencia UML para el proceso de control se muestra en la Figura 23.

El componente GENA implementa el proceso de concurso completo. Incluye las clases:

✎ SubsSolver: se ejecuta en el dispositivo para manejar las solicitudes de sub-cripción desde el punto de control.

✎ EventSolver: se ejecuta en el punto de control para controlar los eventos enviados por el dispositivo.

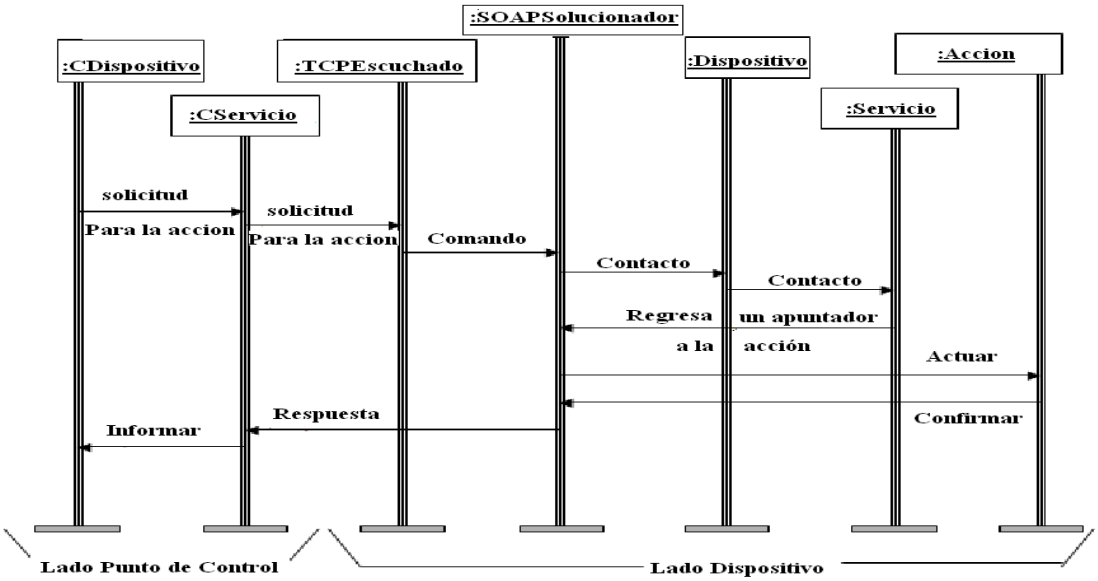


Figura 23. Descripción del proceso de Control

FUENTE: www.win.tue.nl/~johanl/projects/EES5413/UPnP_IASTED.pdf

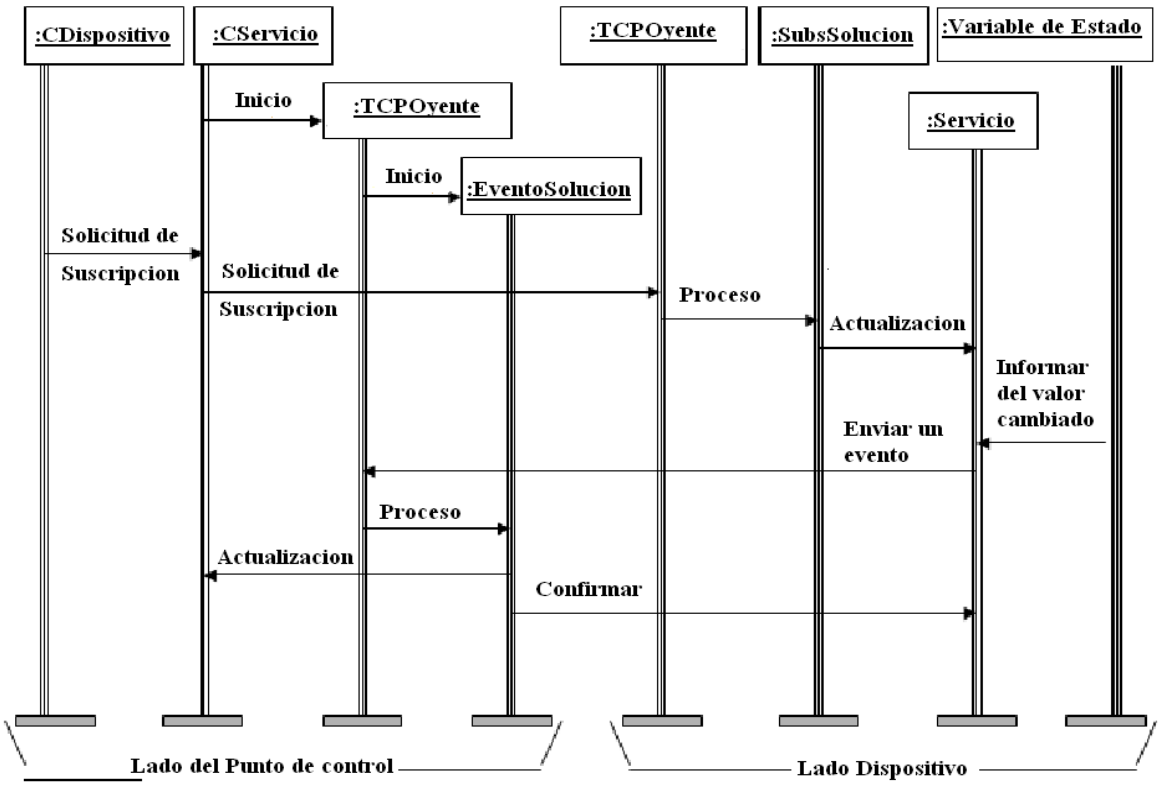


Figura 24. Proceso de concurso Completo

FUENTE: www.win.tue.nl/~johanl/projects/EES5413/UPnP_IASTED.pdf

2.2.7. Sistema en Funcionamiento

La Figura 25 muestra un ejemplo de un dispositivo en funcionamiento. Para ese dispositivo en particular tiene tres servicios, donde los servicios 2 y 3 se asignan en el mismo puerto.

Dentro del tercer servicio se observan las acciones que este proporciona. Asociar cada puerto, es una tarea del servidor que envía los mensajes de los puntos de control para el servicio a que estén destinados. Además de éstos hay otro servidor para hacer frente a los mensajes entrantes UDP para el descubrimiento de

servicios. Todos los servidores en las figuras 25 y 26 usan HTTPe, excepto en el servidor Web.

La relación con el dispositivo real puede ser implementada en gran número de maneras, por ejemplo, a través de una correspondencia directa en el hardware o mediante un dispositivo proxy que se comunica con el dispositivo real a través de algún protocolo propietario. De esta manera, es posible incluir diferentes protocolos al UPnP y dejar que tengan una funcionalidad puente en el nivel funcional.

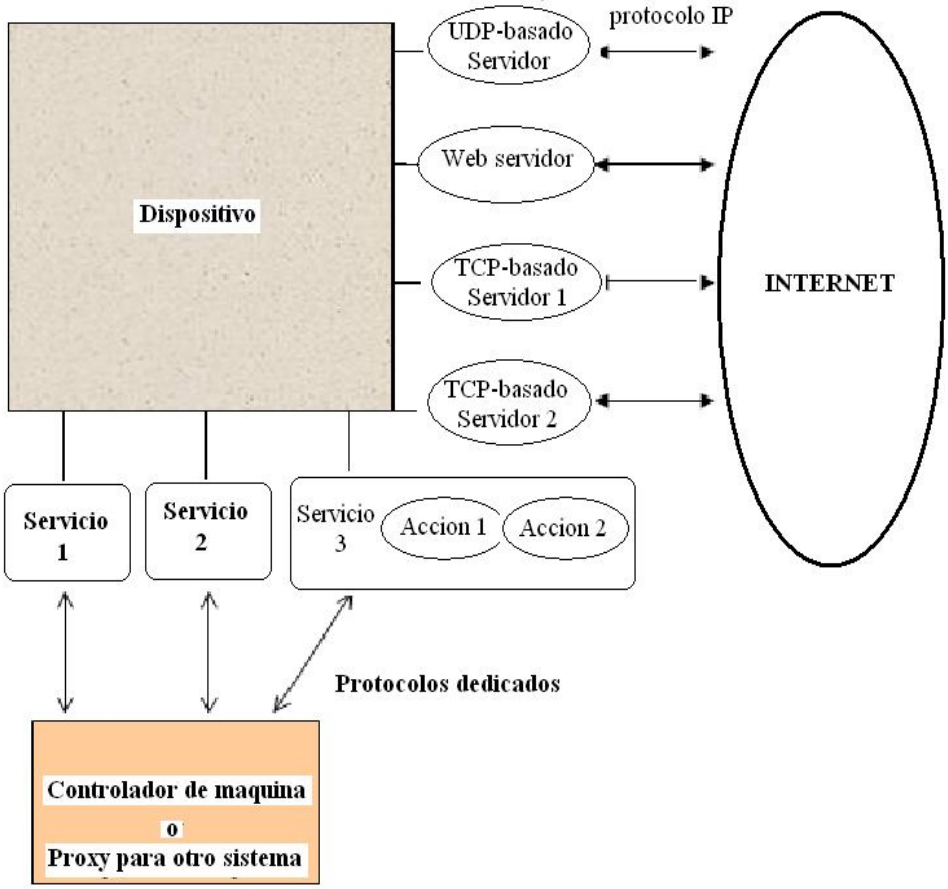


Figura 25. Ejemplo de un Dispositivo funcionando

FUENTE: www.win.tue.nl/~johanl/projects/EES5413/UPnP_IASTED.pdf

La Figura 26 presenta el punto de control en funcionamiento. Dentro de ella los proxies para los dispositivos: CDispositivo1 y CDispositivo2. En el segundo dispositivo se observan los servicios asociados. De nuevo, cada puerto tiene un servidor asociado, en este caso para hacer frente a eventos de notificaciones. CDispositovo2 está controlando el dispositivo de la Figura 25.

Observe que en ambos lados (punto de control y el dispositivo) varios servidores HTTPe deben estar en ejecución para controlar el descubrimiento y los procesos de peer-to-peer.

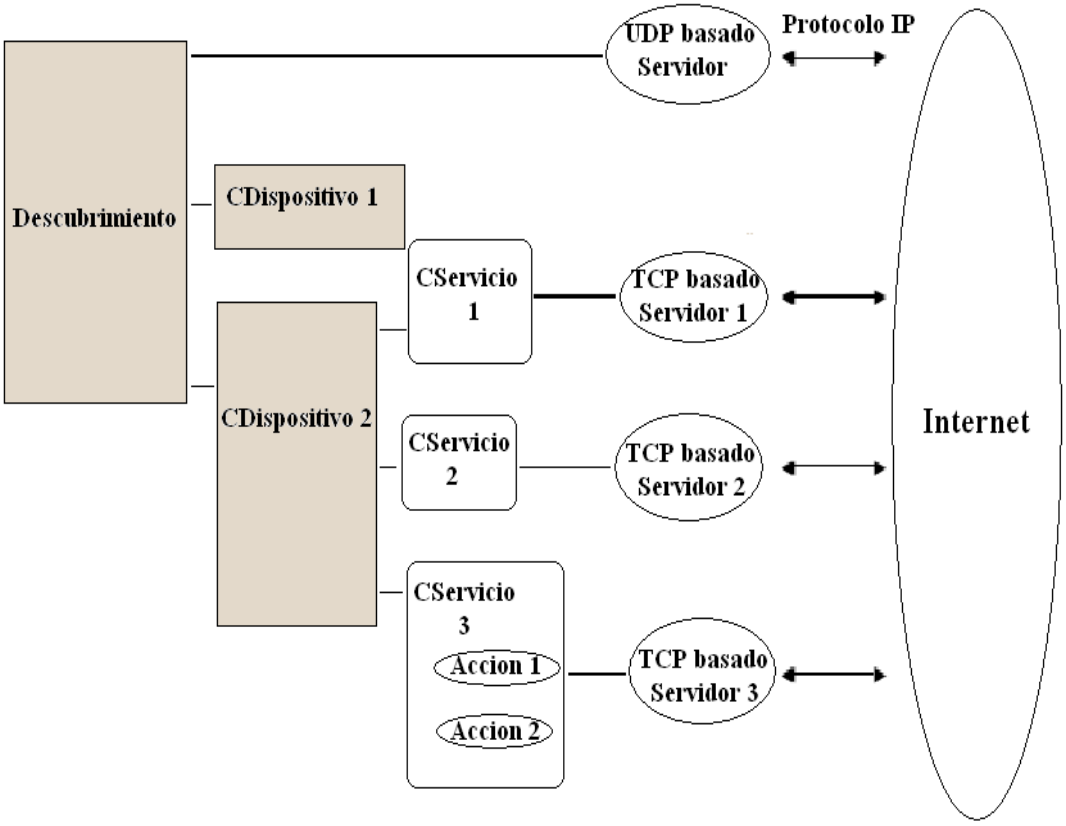


Figura 26. Ejemplo de un punto de control funcionado

FUENTE: www.win.tue.nl/~johanl/projects/EES5413/UPnP_IASTED.pdf

En vista de la limitada potencia de proceso de los sistemas integrados, el número de tareas concurso debe ser limitado. Entonces se propone un modelo para un servidor HTTPe como se muestra en la Figura 27. Un servidor HTTPe mantiene una cola FIFO para todas las solicitudes de los clientes. Así, por puerto, todas las solicitudes se manejan de forma secuencial. A pesar de que selecciona esta opción, hay dos aspectos a tener en cuenta.

✎ Cada petición-respuesta debe ser manejado en un tiempo t lo suficientemente corto como está prescrito por la norma UPnP.

✎ Los tamaños de la cola, deben ser suficientes para que la cola puede contener todas las solicitudes.

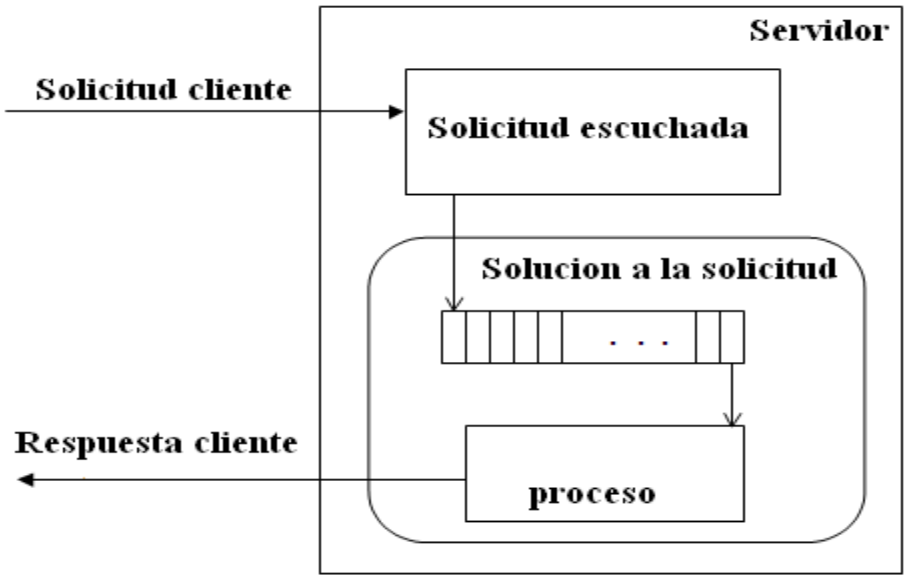


Figura 27. Modelo Servidor HTTPe

FUENTE: www.win.tue.nl/~johanl/projects/EES5413/UPnP_IASTED.pdf

2.2.8. Limitaciones del UPnP

Como toda tecnología, UPnP asume ciertas limitaciones. A continuación se detallan las más
destacadas:

- **Seguridad:** UPnP no dispone por defecto de un mecanismo de autenticación. Sin embargo, sí dispone de DCPs[3], descripciones de servicios, que incorporan políticas de seguridad a la arquitectura UPnP. El servicio de seguridad para dispositivos de UPnP (*UPnP DeviceSecurity Service*) proporciona los servicios necesarios para la autenticación, autorización, prevención de repeticiones y privacidad para las acciones SOAP de UPnP. [6]

- **QoS:** tampoco UPnP por defecto incorpora políticas que aseguren un buen servicio acorde al tipo de datos, cantidad y tiempo necesario para ser transmitidos, especialmente indicadas en transmisiones de voz y video. Aunque, como en el caso anterior, si provee de DCPs que definen una arquitectura para la gestión de políticas de calidad de servicio. [8]

2.2.9. Justificación del uso de UPnP

La realización de este proyecto viene motivada por la implementación de un dispositivo conforme a la especificación UPnP para puntos de control. El UPnP define un conjunto de protocolos para facilitar la comunicación entre entidades conectadas en una misma red sin que el usuario tenga que realizar complicadas configuraciones ni instalar drivers para su uso. Por ese hecho, resulta una

[3] DCPs: Documentos del protocolo de control de dispositivos

tecnología realmente interesante para su aplicación en sistemas de entretenimiento en los que participan varias entidades que permiten compartir el contenido multimedia en la red y el manejo remoto de su reproducción.

Otras de las razones por las cuales se justifica el uso del UPnP son las siguientes:

- **Independencia de medios y dispositivos**: Puede funcionar sobre cualquier medio incluyendo líneas telefónicas, cables de la luz, Ethernet, RF, wireless, y 1394. Esto lo hace apropiado para usos en Domotica.
- **Independencia de Plataformas**: No importa el lenguaje de programación ni el sistema operativo para el desarrollo de productos con esta tecnología.
- **Tecnologías basadas en Internet**: Está desarrollada sobre IP, TCP, UDP, HTTP y XML entre otras.
- **Control de programación**: Ofrece una aplicación convencional de control de programación.
- **Base de protocolos comunes.** Los vendedores están de acuerdo con los protocolos base para la comunicación entre dispositivos.
- **Extensible.** Cada producto UPnP puede agregar servicios de valor en la parte superior de las capas de la arquitectura básica del dispositivo, dependiendo de los distintos fabricantes.

De amplio alcance

La tecnología UPnP es objetivos de las redes domésticas, las redes de proximidad y las redes en las pequeñas empresas y edificios comerciales. Ya que permite la comunicación de datos entre los dispositivos bajo el mando de cualquier dispositivo de control en la red.

Configuración automática desde cero y Descubrimiento

La arquitectura UPnP soporta sin necesidad de configuración y mediante descubrimiento automático las siguientes actividades de un dispositivo:

- Unirse de manera dinámica a una red.
- Obtener una dirección IP.
- Anunciar su nombre.
- Transmitir sus capacidades bajo petición.
- Obtener información sobre la presencia y funciones de otros dispositivos.
- Dejar una red de forma satisfactoria y sin dejar ninguna información de estado no deseado detrás.

Protocolos de control de dispositivos estandarizados

Al igual que la creación de estándares de Internet, la iniciativa UPnP implica una colaboración multi-fabricante para el establecimiento de protocolos estándar de dispositivos de control. Similar a la comunicación basada en Internet, estos se basan en protocolos que son:

- Declarativos.
- Expresado en XML.
- Comunicados a través de HTTP.

2.2.10. Aplicaciones

La tecnología UPnP hace las redes domésticas más sencillas y accesibles, para vivir la experiencia de un hogar conectado a los usuarios y una gran oportunidad para la industria.

Las normas y servicios del dispositivo UPnP han sido definidas y publicadas para gateways de Internet, routers, dispositivos multimedia de audio y video, impresoras, escáneres, control de temperatura, iluminación y puntos de acceso inalámbricos y cámaras digitales de seguridad, y las características avanzadas tales como la seguridad, el usuario remoto interfaz, y la calidad de servicio.

La arquitectura UPnP ofrece conectividad de red extendida entre todos los tipos de dispositivos, incluyendo la red habilitada para los consumidores como son equipos electrónicos, electrodomésticos inteligentes, dispositivos inalámbricos portátiles, PCs, etc. UPnP aprovecha la arquitectura TCP / IP y otras tecnologías de Internet para permitir la integración sin fisuras de estos dispositivos en infraestructuras de red existentes. La tecnología UPnP puede implementarse en cualquier sistema operativo y funciona con cualquier tipo de medios de red física que soporta IP, con cable o inalámbrico proporcionando muchas opciones para el usuario y el desarrollo, que resultan en mayores beneficios económicos para todos.
Entre las aplicaciones del protocolo UPnP se cuenta con los siguientes escenarios, todos ellos planteados por el foro UPnP:

2.2.10.1. Media Server y Procesador de Medios

Usted está compartiendo una barbacoa en la terraza con sus amigos. Alguien recuerda alguna canción favorita de la universidad. Usted comenta que tiene todo el disco en la red. Con tan solo el toque de algunos botones en la pantalla táctil, la canción se escuchara sonando en los altavoces al aire libre.

2.2.10.2. Calidad de Servicio y Media Server

Usted está disfrutando de una película en su Cine en Casa, cuando sus hijos llegan a casa y desean jugar juegos de video. Usted decide desalojar su sala y terminar la película en el dormitorio. UPnP se asegura de que su película tenga

prioridad sobre los videojuegos de los niños y su película se reproduce sin ningún problema hasta su terminación en el clima más hospitalario de su dormitorio.

2.2.10.3. Controles de Iluminación

Después de un viernes por la noche con sus hijos y sus amigos en casa, la casa se vuelve un lío. ¿ Las luces están a la izquierda? ¿Quién sabe si las puertas quedaron abiertas? No hay que preocuparse. Con sólo pulsar un botón, su sistema de alarma descubre que el garaje está abierto. El garaje y persianas diversas se cerrarán automáticamente. El sistema de alarma se activa y todas las luces se apagan.

2.2.10.4. Energía Baja, el Media Server y acceso remoto

Estás visitando a tus padres. Puesto que son los únicos remotamente interesados en las fotos de sus vacaciones, usted insiste en mostrárselas. Desde su decodificador compatible con UPnP habilitado, de forma remota contacta con el servidor en su casa, se activa desde el modo de receso y se levanta usted las imágenes de Acapulco. Un par de horas más tarde, cuando la presentación se haya completado, el servidor (y sus padres) irán a dormir.

2.2.10.5. Procesador de Medios

En medio de un partido de fútbol, su viejo amigo de la secundaria llama por teléfono para conversar. Usted sabe esto porque la persona que llama-ID apareció en la pantalla del televisor. Ya que no lo he visto por un tiempo (y ya que usted tiene UPnP), decide hablar con él. Cuando usted levanta el teléfono, el DVR se detiene para que no se pierda ningún segundo del partido de fútbol. Al colgar, el partido de fútbol continúa donde lo dejo.

2.2.10.6. Impresión

Mientras ve un anuncio en la televisión, un pequeño gráfico aparece para decirles que es un anuncio interactivo que tiene más información. Usted decide activarlo pulsando un botón del mando a distancia. ¡El suéter lindo en el anuncio está a la venta y con descuento! Usted hace clic en el botón otra vez, porque esto es algo que tiene una necesidad imperiosa. Un mensaje de correo electrónico se envía inmediatamente a usted y su impresora en la oficina comienza la impresión del cupón con el 50% de descuento.

2.2.10.7. Contenido de sincronización y de Seguridad

Ya que son totalmente compatibles con UPnP, el cine en casa va a grabar en el DVR en el en el servidor de su coche (filtrado, por supuesto, por su control parental). UPnP hace que sus copias son legítimas marcando los derechos de usuario, garantizando así que usted no sea detenido por la policía por violar el derecho de copyright. En la unidad extendida siguiente, los niños disfrutan de las nuevas grabaciones en el asiento trasero.

2.2.10.8. Gestión de dispositivos

Una noche mientras ve un vídeo de música te das cuenta de que hay colores extraños que no son parte del vídeo, pero que es un problema con su televisor. La TV se da cuenta de esto y de inmediato contacta a soporte al cliente. Un mensaje aparece en la pantalla y la persona de servicio en el otro extremo pide permiso para arreglar el problema. Utiliza su control remoto para otorgar el permiso. La solución se descarga en la TV y el problema se resuelve.

2.3. DISPOSITIVOS UPnP

2.3.1. Definición

Los dispositivos UPnP son productos tecnológicos que dentro de su diseño y configuración manejan el protocolo UPnP, que son usados para el hogar, la oficina y cualquier otro lugar donde se necesiten, un ejemplo de estos son: las impresoras y celulares de última generación que traen incluido este servicio de comunicación. Las empresas que fabrican estos dispositivos conforman una organización llamada “Fórum UPnP “en donde hacen conocer los productos y aportan nuevas ideas para la estandarización de comunicación entre todos los dispositivos disponibles por los fabricantes.

2.3.2. Aplicaciones

La arquitectura UPnP ofrece conectividad omnipresente red peer-to-peer de los PCs de todos los factores de forma, aparatos inteligentes y dispositivos inalámbricos. La arquitectura UPnP es una arquitectura distribuida, abierta de red que aprovecha TCP / IP y la Web para habilitar la red de proximidad sin fisuras, además de control y transferencia de datos entre los dispositivos conectados en red en el hogar, oficina y cualquier otro lugar.

Dispositivos Inteligentes

La tecnología informática ahora es bastante barata ya que las organizaciones pueden construir la inteligencia en el diseño de muchos dispositivos electrónicos comunes en la actualidad. Bajo costo, la creación de redes fiables, junto con una nueva clase de dispositivos de computación pequeños, está creando oportunidades para nuevos productos y funcionalidades basadas principalmente en la conectividad natural y transparente entre estos dispositivos.

La conectividad ahora se utiliza para:

• Control de dispositivos de forma remota

• Mover datos digitales en forma de audio, vídeo e imágenes fijas entre

dispositivos

• Compartir información entre dispositivos y con la World Wide Web.

2.4. TECNOLOGIAS APLICADAS

2.4.1. Sistema Operativo Windows XP

2.4.1.1. Definición

Windows XP es una versión de Microsoft Windows, la línea de sistemas operativos desarrollado por Microsoft. Lanzado al mercado el 25 de octubre de 2001, actualmente es el sistema operativo para x86 más utilizado del planeta y se considera que existen más de 400 millones de copias funcionando. Las letras "**XP**" provienen de la palabra *e**XP**eriencia*.

Dispone de versiones para varios entornos informáticos, incluyendo PCs domésticos o de negocios, equipos portátiles, "netbooks", "tablet PC" y "media center". Sucesor de Windows 2000 junto con Windows ME, y antecesor de Windows Vista, es el primer sistema operativo de Microsoft orientado al consumidor que se construye con un núcleo y arquitectura de Windows NT disponible en versiones para plataformas de 32 y 64 bits.

A diferencia de versiones anteriores de Windows, al estar basado en la arquitectura de Windows NT proveniente del código de Windows 2000, presenta mejoras en la estabilidad y el rendimiento. Tiene una interfaz gráfica de usuario (GUI) perceptiblemente reajustada (denominada *Luna*), la cual incluye características rediseñadas, algunas de las cuales se asemejan ligeramente a otras GUI de otros sistemas operativos, cambio promovido para un uso más fácil que en las versiones anteriores. Se introdujeron nuevas capacidades de gestión de software para evitar el "*DLL Hell*" (*infierno de las DLLs*) que plagó las viejas

versiones. Es también la primera versión de Windows que utiliza la activación del producto para reducir la piratería del software, una restricción que no sentó bien a algunos usuarios. Ha sido también criticado por las vulnerabilidades de seguridad, integración de Internet Explorer, la inclusión del reproductor Windows Media Player y aspectos de su interfaz.

2.4.1.2. Desarrollo

El desarrollo de Windows XP parte desde la forma de Windows Neptune. Windows XP fue desarrollado en 18 meses, desde diciembre de 1999 hasta agosto de 2001.

Microsoft producía dos líneas separadas de sistemas operativos. Una línea estaba dirigida a las computadoras domésticas basada en un núcleo de MS-DOS y representada por Windows 95, Windows 98 y Windows Me, mientras que la otra, basada en un Núcleo "NT" es representada por Windows NT y Windows 2000, estaba pensada para el mercado corporativo y empresarial e incluía versiones especiales para servidores. Windows ME "Millenium" fue un intento por parte de Microsoft de ofrecer un único sistema operativo multiuso, aunque falló por poseer el núcleo de arranque de MS-DOS con el código NT de Windows, Windows XP fue la verdadera fusión de un sistema operativo único basado enteramente en la arquitectura NT contando con la funcionalidad de MS-DOS, con él, se eliminó definitivamente el soporte para los programas basados en MS-DOS del sistema operativo.

2.4.1.3. Características

Windows XP introdujo nuevas características

- Ambiente gráfico
- Secuencias más rápidas de inicio y de hibernación.

- Capacidad del sistema operativo de desconectar un dispositivo externo, de instalar nuevas aplicaciones y controladores sin necesidad de reiniciar.
- Una nueva interfaz de uso más fácil, incluyendo herramientas para el desarrollo de temas de escritorio.
- Uso de varias cuentas, lo que permite que un usuario guarde el estado actual y aplicaciones abiertos en su escritorio y permita que otro usuario abra una sesión sin perder esa información.
- ClearType, diseñado para mejorar legibilidad del texto encendido en pantallas de cristal líquido (LCD) y monitores similares.
- Escritorio Remoto, que permite a los usuarios abrir una sesión con una computadora que funciona con Windows XP a través de una red o Internet, teniendo acceso a sus usos, archivos, impresoras, y dispositivos;
- Soporte para la mayoría de módems ADSL y conexiones wireless, así como el establecimiento de una red FireWire.

2.4.2. Lenguaje de desarrollo: Visual Studio C#

2.4.2.1. Definición

C# es un lenguaje orientado a objetos elegante y con seguridad de tipos que permite a los desarrolladores crear una amplia gama de aplicaciones sólidas y seguras que se ejecutan en .NET Framework. Puede utilizar este lenguaje para crear aplicaciones cliente para Windows tradicionales, servicios Web XML, componentes distribuidos, aplicaciones cliente-servidor, aplicaciones de base de datos, y muchas tareas más. Microsoft Visual C# 2008 proporciona un editor de código avanzado, diseñadores de interfaz de usuario prácticos, un depurador integrado y muchas otras herramientas para facilitar un rápido desarrollo de la aplicación basado en la versión 2.0 del lenguaje C# y en .NET Framework.

2.4.2.2. Características

Con la idea de que los programadores más experimentados puedan obtener una visión general del lenguaje, a continuación se recoge de manera resumida las principales características de C#. Alguna de las características aquí señaladas no son exactamente propias del lenguaje sino de la plataforma .NET en general. Sin embargo, también se comentan aquí también en tanto que tienen repercusión directa en el lenguaje, aunque se indicará explícitamente cuáles son ese tipo de características cada vez que se toquen:

- Sencillez: C# elimina muchos elementos que otros lenguajes incluyen y que son innecesarios en .NET. Por ejemplo:
  - El código escrito en C# es autocontenido, lo que significa que no necesita de ficheros adicionales al propio fuente tales como ficheros de cabecera o ficheros IDL.
  - El tamaño de los tipos de datos básicos es fijo e independiente del compilador, sistema operativo o máquina para quienes se compile (no como en C++), lo que facilita la portabilidad del código.
  - No se incluyen elementos poco útiles de lenguajes como C++ tales como macros, herencia múltiple o la necesidad de un operador diferente del punto (.) acceder a miembros de espacios de nombres (::)
- Modernidad: C# incorpora en el propio lenguaje elementos que a lo largo de los años ha ido demostrándose son muy útiles para el desarrollo de aplicaciones y que en otros lenguajes como Java o C++ hay que simular, como un tipo básico decimal que permita realizar operaciones de alta precisión con reales de 128 bits (muy útil en el mundo financiero), la inclusión de una instrucción foreach que permita recorrer colecciones con facilidad y es ampliable a tipos definidos por el usuario, la inclusión de un

tipo básico string para representar cadenas o la distinción de un tipo bool específico para representar valores lógicos.

- Orientación a objetos: Como todo lenguaje de programación de propósito general actual, C# es un lenguaje orientado a objetos, aunque eso es más bien una característica del CTS que de C#. Una diferencia de ese enfoque orientado a objetos respecto al de otros lenguajes como C++ es que el de C# es más puro en tanto que no admiten ni funciones ni variables globales sino que todo el código y datos han de definirse dentro de definiciones de tipos de datos, lo que reduce problemas por conflictos de nombres y facilita la legibilidad del código.

- Orientación a componentes: La propia sintaxis de C# incluye elementos propios del diseño de componentes que otros lenguajes tienen que simular mediante construcciones más o menos complejas. Es decir, la sintaxis de C# permite definir cómodamente propiedades (similares a campos de acceso controlado), eventos (asociación controlada de funciones de respuesta a notificaciones) o atributos (información sobre un tipo o sus miembros).

- Gestión automática de memoria: Como ya se comentó, todo lenguaje de .NET tiene a su disposición el recolector de basura del CLR. Esto tiene el efecto en el lenguaje de que no es necesario incluir instrucciones de destrucción de objetos. Sin embargo, dado que la destrucción de los objetos a través del recolector de basura es indeterminista y sólo se realiza cuando éste se active –ya sea por falta de memoria, finalización de la aplicación o solicitud explícita en el fuente-, C# también proporciona un mecanismo de liberación de recursos determinista a través de la instrucción using.

- Seguridad de tipos: C# incluye mecanismos que permiten asegurar que los accesos a tipos de datos siempre se realicen correctamente, lo que permite

evita que se produzcan errores difíciles de detectar por acceso a memoria no perteneciente a ningún objeto y es especialmente necesario en un entorno gestionado por un recolector de basura.

- Instrucciones seguras: Para evitar errores muy comunes, en C# se han impuesto una serie de restricciones en el uso de las instrucciones de control más comunes. Por ejemplo, la guarda de toda condición ha de ser una expresión condicional y no aritmética, con lo que se evitan errores por confusión del operador de igualdad (==) con el de asignación (=); y todo caso de un switch ha de terminar en un break o goto que indique cuál es la siguiente acción a realizar, lo que evita la ejecución accidental de casos y facilita su reordenación.

- Sistema de tipos unificado: A diferencia de C++, en C# todos los tipos de datos que se definan siempre derivarán, aunque sea de manera implícita, de una clase base común llamada System.Object, por lo que dispondrán de todos los miembros definidos en ésta clase (es decir, serán “objetos”).

- Extensibilidad de tipos básicos: C# permite definir, a través de estructuras, tipos de datos para los que se apliquen las mismas optimizaciones que para los tipos de datos básicos. Es decir, que se puedan almacenar directamente en pila (luego su creación, destrucción y acceso serán más rápidos) y se asignen por valor y no por referencia. Para conseguir que lo último no tenga efectos negativos al pasar estructuras como parámetros de métodos, se da la posibilidad de pasar referencias a pila a través del modificador de parámetro ref.

- Extensibilidad de operadores: Para facilitar la legibilidad del código y conseguir que los nuevos tipos de datos básicos que se definan a través de las estructuras estén al mismo nivel que los básicos predefinidos en el lenguaje, al igual que C++ y a diferencia de Java, C# permite redefinir el significado de la mayoría de los operadores -incluidos los de conversión,

tanto para conversiones implícitas como explícitas- cuando se apliquen a diferentes tipos de objetos.

- Versionable: C# incluye una política de versionado que permite crear nuevas versiones de tipos sin temor a que la introducción de nuevos miembros provoquen errores difíciles de detectar en tipos hijos previamente desarrollados y ya extendidos con miembros de igual nombre a los recién introducidos.

- Eficiente: En principio, en C# todo el código incluye numerosas restricciones para asegurar su seguridad y no permite el uso de punteros. Sin embargo, y a diferencia de Java, en C# es posible saltarse dichas restricciones manipulando objetos a través de punteros. Para ello basta marcar regiones de código como inseguras (modificador unsafe) y podrán usarse en ellas punteros de forma similar a cómo se hace en C++, lo que puede resultar vital para situaciones donde se necesite una eficiencia y velocidad procesamiento muy grandes.

- Compatible: Para facilitar la migración de programadores, C# no sólo mantiene una sintaxis muy similar a C, C++ o Java que permite incluir directamente en código escrito en C# fragmentos de código escrito en estos lenguajes, sino que el CLR también ofrece, a través de los llamados Platform Invocation Services (PInvoke), la posibilidad de acceder a código nativo escrito como funciones sueltas no orientadas a objetos tales como las DLLs de la API Win32. Nótese que la capacidad de usar punteros en código inseguro permite que se pueda acceder con facilidad a este tipo de funciones, ya que éstas muchas veces esperan recibir o devuelven punteros.

2.4.2.3. Ventajas

Ventajas frente a C y C++

- Compila a código intermedio (CIL) independiente del lenguaje en que haya sido escrita la aplicación e independiente de la máquina donde vaya a ejecutarse
- Recolección de basura automática
- Eliminación del uso de punteros, en C# no se necesitan
- No hay que preocuparse por archivos de cabecera ".h"
- No importa el orden en que hayan sido definidas las clases ni las funciones
- No hay necesidad de declarar funciones y clases antes de definirlas
- No existen las dependencias circulares
- Soporta definición de clases dentro de otras
- No existen funciones, ni variables globales, todo pertenece a una clase
- Todos los valores son inicializados antes de ser usados (automáticamente se inicializan al valor estandarizado, o manualmente se pueden inicializar desde constructores estáticos)
- No se pueden utilizar valores no booleanos (enteros, coma flotante...) para condicionales. Es mucho más limpio y menos propenso a errores
- Puede ejecutarse en una sandbox restringida

Ventajas frente a C++ y java

- Concepto formalizado de los métodos get y set, con lo que se consigue código mucho más legible

- Gestión de eventos (usando delegados) mucho más limpia

Ventajas frente a java

- El rendimiento es, por lo general, mucho mejor
- CIL (el lenguaje intermedio de .NET) está estandarizado, mientras que los bytecodes de java no lo están
- Soporta bastantes más tipos primitivos (*value types*), incluyendo tipos numéricos sin signo
- Indizadores que permiten acceder a cualquier objeto como si se tratase de un array
- Compilación condicional
- Aplicaciones multi-hilo simplificadas
- Soporta la sobrecarga de operadores, que aunque pueden complicar el desarrollo son opcionales y algunas veces muy útiles
- Permite el uso (limitado) de punteros cuando realmente se necesiten, como al acceder a librerías nativas que no se ejecuten sobre la máquina virtual

2.4.2.4. Requerimientos del sistema

- Equipo con un procesador de 600 MHZ o más rápido.
- 192 MG de RAM o más.
- 2 GB de espacio disponible en disco duro.
- Resolución de pantalla de 1024x768 o superior con 256 colores.

- El sistema operativo XP debe contener Service Pack 2 u versiones posteriores.

2.4.3. NET FRAMEWORK 3.5

2.4.3.1. Definición

El Microsoft .NET Framework, es un componente de software que puede ser o es incluido en los sistemas operativos Microsoft Windows. Provee soluciones pre-codificadas para requerimientos comunes de los programas y gestiona la ejecución de programas escritos específicamente para este Framework.

Microsoft desea que todas las aplicaciones creadas para la plataforma Windows, sean basadas en el .NET Framework. Su objetivo es crear un marco de desarrollo de software sencillo, reduciendo las vulnerabilidades y aumentando la seguridad de los programas desarrollados.

Las soluciones pre-codificadas que forman la biblioteca .NET, cubren un gran rango de necesidades de la programación de programas. Los programadores las emplean y combinan con sus propios códigos en sus programas. El Framework incluye soluciones en áreas como: la interfaz de usuario, acceso a datos, conectividad a bases de datos, criptografía, desarrollo de aplicaciones web, algoritmos numéricos y comunicación de redes.
Con esta plataforma Microsoft incursiona de lleno en el campo de los servicios web y establece al XML como norma en el transporte de información en sus productos y lo promociona como tal en los sistemas desarrollados utilizando sus herramientas.

2.4.3.2. Características

* Integración total de LINQ (Language Integrated Query) y del reconocimiento de los datos. Esta nueva característica le permitirá escribir código en idiomas habilitados para LINQ para filtrar, enumerar y crear proyecciones de varios tipos de datos SQL, colecciones, XML y conjuntos de datos usando la misma sintaxis.

* ASP.NET AJAX le permite crear experiencias web más eficaces, más interactivas y con un gran índice de personalización que funcionan con los exploradores más usados.

* Nueva compatibilidad con el protocolo web para generar servicios WCF, como por ejemplo AJAX, JSON, REST, POX, RSS, ATOM y distintos estándares WS-* nuevos.

* Compatibilidad total con las herramientas de Visual Studio 2008 para WF, WCF y WPF, incluida la nueva tecnología de servicios habilitados para flujos de trabajo.

* Nuevas clases en la biblioteca de clases base (BCL) de .NET Framework 3.5 que tratan numerosas solicitudes de cliente comunes.

2.4.3.3. Ventajas

Las principales ventajas de la utilización de un Framework son:

1. El desarrollo rápido de aplicaciones. Los componentes incluidos en un Framework constituyen una capa que libera al programador de la escritura de código de bajo nivel.

2. La reutilización de componentes software al por mayor. Los Frameworks son los paradigmas de la reutilización.

3. El uso y la programación de componentes que siguen una política de diseño uniforme. Un Framework orientado a objetos logra que los componentes sean clases que pertenezcan a una gran jerarquía de clases, lo que resulta en bibliotecas más fáciles de aprender a usar.

Las desventajas de los Frameworks son:

1. La dependencia del código fuente de una aplicación con respecto al Framework. Si se desea cambiar de Framework, la mayor parte del código debe reescribirse.

2. La demanda de grandes cantidades de recursos computacionales debido a que la característica de reutilización de los Frameworks tiende a generalizar la funcionalidad de los componentes. El resultado es que se incluyen características que están "de más", provocando una sobrecarga de recursos que se hace más grande en cuanto más amplio es el campo de reutilización.

2.4.4. NETWORK LIGHT

2.4.4.1. Definición

Esta herramienta de UPnP desarrollada por la empresa INTEL, simula una bombilla, la cual soporta la función de apagado y encendido, además de la opción de dimmer, es decir, cambiar la intensidad luminosa. Cuando se activa dicho dispositivo aparece la foto de una bombilla, la cual se puede ver en la imagen 28.

Figura 28. Herramienta Network Light de Intel

FUENTE: Autores, pantallazo de la aplicación desarrollada por INTEL

2.4.4.2. Características

Este dispositivo UPnP tiene dos servicios, uno llamado DimmingService y el otro SwitchPower. El servicio que se va a usar en la Interfaz Gráfica es SwitchPower, el cual tiene dos funciones, GetStatus y SetTarget.

La función GetStatus devuelve un valor, el cual puede ser 0 ó 1, lo que indica que la luz está apagada o encendida respectivamente.

2.4.4.3. Aplicación

La función que utiliza el software del Gestor de Dispositivos UPnP para encender o apagar la luz es SetTarget, la cual contiene una variable llamada newTargetValue, que cuando se pone a true y se invoca dicha función enciende la bombilla. Si se pone a false, se apagará la luz.

3. IMPLEMENTACION

En el actual capítulo se aborda el diseño del Gestor de Dispositivo UPnP en base a los criterios establecidos por las distintas tecnologías que se han ido definiendo en el capitulo anterior.

Para facilitar la compresión del capítulo se han utilizado varios diagramas UML que sintetizan de forma gráfica las características y arquitectura del sistema diseñado.

Como paso previo se analizaron los requisitos del sistema, como resultado del análisis se definieron los Casos de Uso para la aplicación.

3.1. Especificación de requisitos

En primer lugar se ha de analizar los requisitos previos que se cumplirán, los cuales son:

1. Permitir al usuario abrir o cerrar los puertos necesarios para la comunicación UPnP.

2. Permitir al usuario realizar en la red una búsqueda particular sobre algún tipo específico de dispositivo UPnP y su posterior conexión con este.

3. Ofrecer al usuario la opción de observar el estado actual de la búsqueda e intercambio de mensajes entre dispositivos y el Gestor UPnP además de los cambios que surjan durante estos procesos.

4. Mostrar al usuario el estado actual de las propiedades de los dispositivos UPnP seleccionados.

5. Presentar al usuario una lista con la descripción básica del dispositivo UPnP detectado.

6. Presentar al usuario una lista con los posibles servicios que se puede utilizar según el dispositivo UPnP encontrado.

7. Permitir al usuario detener o iniciar la búsqueda de dispositivos en cualquier momento.

3.1.1. Casos de Uso

Tras haber definido los requisitos del sistema se pude generar el Diagrama de casos de Uso. Este diagrama pretende representar de forma gráfica y sencilla la funcionalidad de la aplicación y su interacción con los actores que intervienen en el proceso.

A continuación se presenta el diagrama y se describen brevemente los casos de uso del mismo:

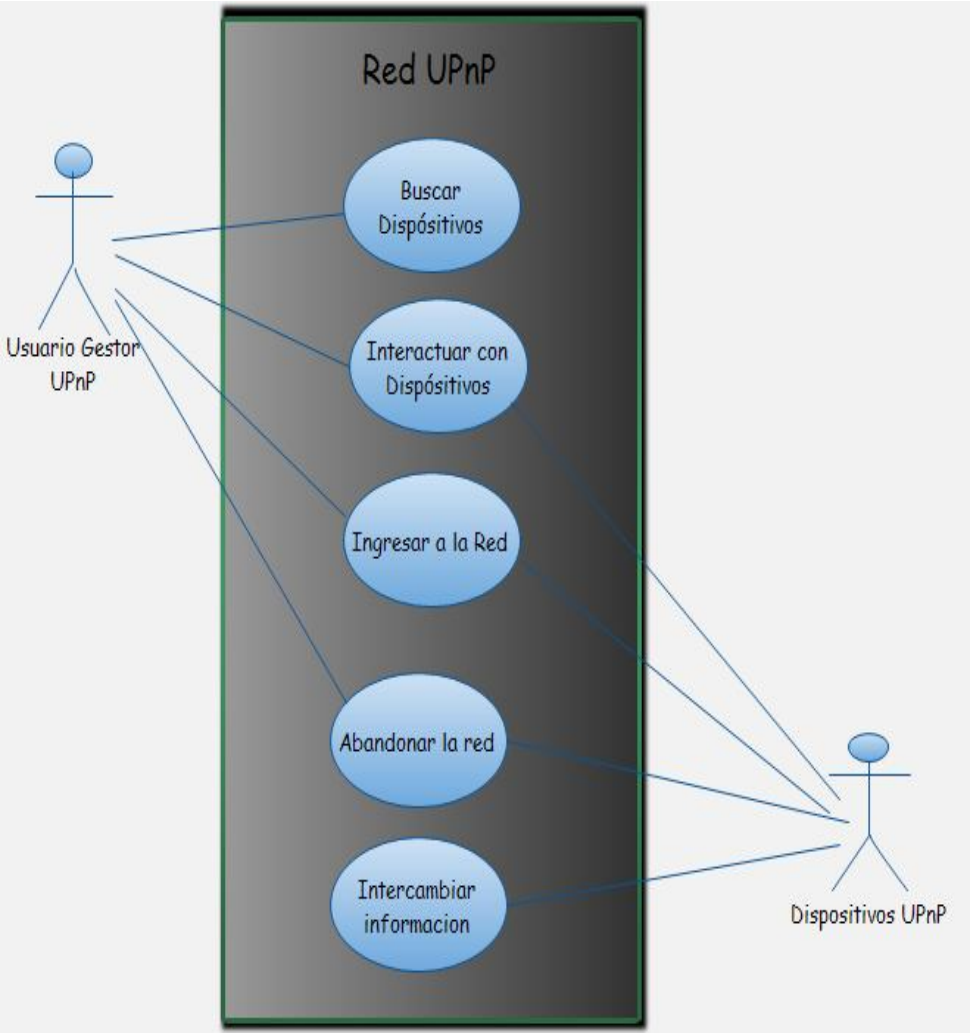


Figura 29. Diagrama de Casos de Uso del Gestor UPnP

FUENTE: Autores

Se han definido dos actores diferentes:

Gestor UPnP (Usuario): se trata del *Control Point* de la arquitectura UPnP para dispositivos. El cual permite al usuario controlar los dispositivos además de recibir la información de estado de los servicios asociados a éste.

Dispositivos UPnP: se trata de los dispositivos de UPnP disponibles en la red. Proporcionan el contenido del fabricante y sus funciones que se comparte en la red UPnP.

Descripción casos de uso:

Nombre:	Interactuar con Dispositivos
Descripción:	Controlar por medio del gestor UPnP los servicios disponibles de un dispositivo UPnP en particular conectado en red, además de acceder a información sobre este.
Actores:	Gestor UPnP, Dispositivo UPnP
Precondiciones:	Búsqueda del dispositivo
Flujo Normal:	

GESTOR UPnP	**DISPOSITIVO UPnP**
1. Selección del dispositivo con el cual se va a interactuar.	
2. Seleccionar el tipo de acción a ejecutar.	
3. Ejecutar la acción en cuestión.	
	4. Reconocer y aceptar la acción.
	5. Enviar los resultados de la acción.
6. Mostrar los resultados obtenidos.	

Flujo Alternativo: En el caso que la acción solicitada no sea reconocida por el dispositivo, mostrar el mensaje de error para orientar al usuario sobre el problema ocurrido.
Postcondiciones:

Nombre:	Realizar Búsqueda de Dispositivos
Descripción:	Comienza la búsqueda de dispositivos UPnP disponibles dentro de la red implementada, ya sea una búsqueda genérica de todos los tipos de dispositivos o una búsqueda especializada de un dispositivo en particular.
Actores:	Gestor UPnP, Dispositivos UPnP.
Precondiciones:	Encender los dispositivos y habilitar la funcionalidad UPnP.
Flujo Normal:	

GESTOR UPnP	**DISPOSITIVO UPnP**
1. Iniciar el Gestor UPnP.	
2. Seleccionar el tipo de búsqueda a realizar.	
3. Iniciar la búsqueda.	
4. Enviar mensajes de búsqueda.	
	5. Responder mensajes de búsqueda.
6. Mostrar resultados de la búsqueda ya sea exitosa o no se hallen dispositivos.	

Flujo Alternativo: En el momento de mostrar los resultados de la búsqueda aunque no se encuentren dispositivos, esta no se da por terminada sino que continua indefinidamente hasta que se termine de forma explícita.
Postcondiciones:

Nombre:	Agregarse a la red UPnP
Descripción:	Intercambio de mensajes de búsqueda y de aceptación.
Actores:	Gestor UPnP, Dispositivo UPnP
Precondiciones:	Encender los dispositivos y habilitar la funcionalidad UPnP.
Flujo Normal:	

GESTOR UPnP	**DISPOSITIVO UPnP**
1. Desbloquear los puertos de comunicación del protocolo UPnP en el firewall de Windows.	1. Desbloquear los puertos de comunicación del protocolo UPnP en el firewall de Windows.
2. Activar el Gestor UPnP.	2. Activar la aplicación UPnP.
3. Envía mensajes de descubrimiento a la red UPnP.	3. Envía mensajes de Anunciamiento a la red o responde a un mensaje de descubrimiento del gestor.
4. Reconoce el mensaje de aviso o de respuesta al descubrimiento y lo procesa.	
5. Recepciona el mensaje de respuesta del dispositivo y lo agrega a la lista de encontrados.	

Flujo Alternativo: En caso de no encontrar ningún dispositivo en la red, el gestor UPnP continua enviando mensajes de búsqueda periódicamente con lo que permite detectar automáticamente la aparición de un nuevo dispositivo sin necesidad de reiniciar la búsqueda.

Postcondiciones: Mantener activo el proceso de notificación de eventos en caso de que el dispositivo se desconecte de repente.

Nombre:	Abandonar la red UPnP
Descripción:	Intercambio de mensajes de notificación de eventos.
Actores:	Gestor UPnP, Dispositivo UPnP
Precondiciones:	Estar conectado a la red UPnP
Flujo Normal:	

GESTOR UPnP	**DISPOSITIVO UPnP**
	1. Envía un mensaje de cambio de estado a no disponible y sale de la red.
2. Detecta el mensaje de cambio de estado y lo procesa.	
3. Elimina de la lista de Encontrados el Dispositivo en cuestión.	
4. Actualiza la lista de dispositivos Disponibles.	

Flujo Alternativo: Cuando el que abandona la red UPnP es el gestor, no se envían mensajes de notificación.
Postcondiciones:

Nombre:	Intercambiar información
Descripción:	Los dispositivos generan la información necesaria cuando ocurre un cambio en alguna de sus variables de estado, y los envían en mensajes de notificación los cuales son procesados por el Gestor UPnP para actualizar la información sobre cada dispositivo.
Actores:	Gestor UPnP, Dispositivo UPnP
Precondiciones:	Es necesario que el dispositivo se haya agregado con éxito a la red.
Flujo Normal:	

GESTOR UPnP	**DISPOSITIVO UPnP**
	1. Genera información de cambio de estado y envía el mensaje de notificación.
2. Rastrea periódicamente la red en espera de notificaciones.	
3. Recibe y analiza el estado de un dispositivo según el mensaje enviado.	
4. Actualiza la información mostrada al usuario sobre el dispositivo en cuestión.	

Flujo Alternativo: El gestor envía un mensaje de control a un Dispositivo provocando un cambio en sus variables de estado, para luego proseguir con el flujo de notificación de resultados anterior.
Postcondiciones:

3.2. IMPLEMENTACION DE LA APLICACIÓN

En este apartado se presenta la aplicación que ha sido desarrollada, se hará una descripción del proceso de su implementación. Posteriormente se enunciarán las características funcionales principales de la misma, seguido de una síntesis de su arquitectura de clases y métodos. Finalmente, se muestran los diagramas de secuencia de los procesos de mayor relevancia entre todos los que suceden durante su ejecución.

3.2.1. Proceso de implementación de la aplicación.

En las próximas líneas se describen las fases que han compuesto el proceso de implementación de la aplicación. Estas son:

3.2.1.1. Configuración del UPnP en Windows XP

La tecnología UPnP se apoya en los siguientes sistemas de la familia de Windows: Millennium Edition, CE. NET, Windows XP y versiones posteriores .La aplicación en Windows XP relacionada con el entorno UPnP se basa en dos componentes básicos, los servicios del sistema y las bibliotecas COM. Los servicios del sistema relacionados con UPnP son "Universal Plug and Play de acogida de dispositivos" (UPnPHost) y "Servicio de descubrimientos SSDP" (SSDPSRV). Como pudo ver, en el sistema son dos servicios diferentes relacionados con UPnP; esto es porque Microsoft divide las funciones relacionadas con los dispositivos UPnP de alojamiento que prestan los servicios, y funciones para el descubrimiento y el control de dispositivos UPnP.

El servicio UPnPHost, permite en general, ejecutar aplicaciones que desempeñan el papel de los dispositivos UPnP y les da a estos la funcionalidad necesaria compatible con la arquitectura UPnP. Así pues, si desea ejecutar la aplicación del dispositivo UPnP, primero debe verificar si el servicio UPnPHost se está ejecutando. Por el contrario, el servicio SSDPSRV le permite encontrar los dispositivos UPnP en todas las redes disponibles en el sistema. Este servicio se implementa a través del protocolo SSDP (Simple Service Discovery Protocol) y detecta los dispositivos UPnP conectados a la red además notifica al cliente cuando el dispositivo está conectado o desconectado.

En Windows XP, hay pocas interfaces de programación que permiten el uso y desarrollo de la tecnología UPnP en sus aplicaciones. El Host de dispositivo y los puntos de control, como en el caso de los servicios del sistema, son dos interfaces independientes diseñados para la programación en diferentes campos. El API de Host de dispositivo proporciona apoyo a la creación de aplicaciones que se encuentran en Windows y en calidad de dispositivos UPnP, que están exhibiendo sus servicios a otros dispositivos o aplicaciones de tipo punto de control. El API de Host de dispositivo ayuda al desarrollador a crear la funcionalidad básica del dispositivo que incluya los servicios compartidos, e integrarlos con el resto de las funciones necesarias (descubrir, control y eventos), que son proporcionados por entorno UPnP y los servicios del sistema: UPnPHost y SSDPSRV.

Para que la aplicación UPnP funcione en Windows XP, además de los servicios del sistema, también es necesario configurar correctamente la aplicación cortafuegos. SSDP (Protocolo de servicio de SSDPSRV) utiliza el puerto UDP 1900 para la difusión y el puerto TCP 2869 para la notificación de eventos. Por lo tanto, para garantizar el servicio SSDPSRV, y la aplicación del punto de control, en el firewall se deben desbloquear los puertos 1900 UDP y TCP 2869.

3.2.1.2. Definición de que es un punto de control (Gestor UPnP)

El API de punto de control UPnP se utiliza para crear aplicaciones capaces de descubrir y controlar dispositivos UPnP. Contando con los dispositivos, estos se pueden conectar a cualquier tipo de soporte de red física, a la que tiene acceso un equipo que está ejecutando la aplicación. La arquitectura UPnP proporciona a la aplicación todos los mecanismos necesarios para encontrar y controlar los dispositivos UPnP, así como los mecanismos relacionados con la notificación sobre los eventos generados por el dispositivo y la conexión a los dispositivos o desconectar de la red. Un elemento esencial de la arquitectura UPnP que entrega estos mecanismos es un SSDPSRV servicio del sistema. Las clases, a que la API de Punto de Control d se refiere, se almacenan en una biblioteca COM llamada UPnP.dll. Para utilizar esta biblioteca a la perfección vale la pena al principio instalar SDK de la plataforma, donde se necesitan los archivos de cabecera y bibliotecas.

3.2.1.3. Diseño basado en objetos para la API

Para el uso del API de punto de control se idearon tres objetos básicos estrechamente relacionados con la arquitectura de dispositivos UPnP:

• Objeto Buscador de dispositivos

• Objeto Dispositivo,

• Objeto Servicio.

El objeto Buscador de dispositivos lleva a cabo las tareas relacionadas con el descubrimiento de dispositivos en la red y proporcionar los objetos Dispositivo encontrados a la aplicación. El objeto Dispositivo representa un modelo de dispositivo UPnP, que se relaciona directamente con el nivel de arquitectura de

dispositivos UPnP desarrollada por la organización Foro UPnP. Desde la perspectiva de un programador el objeto de dispositivo es un dispositivo UPnP físico, con el que una aplicación se comunica para utilizar sus servicios. El objeto servicio es inseparable y es un elemento esencial de un objeto de dispositivo y su misión es proveer servicios a los clientes (dispositivos, los puntos de mando). Para utilizar los servicios del dispositivo se basa en la invocación por el punto de control de las llamadas "acciones" en el objeto de servicio. Un objeto de dispositivo puede contener muchos objetos de servicio.

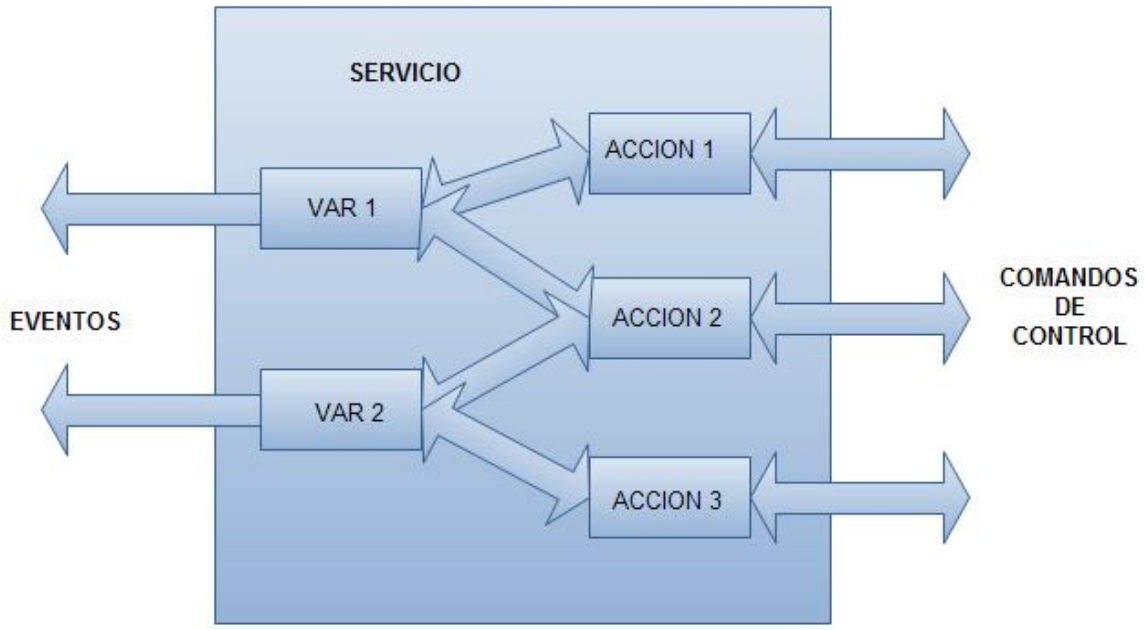


Figura 30. Conformación de los servicios de un dispositivo UPnP

FUENTE: Autores

El objeto servicio, que muestra la imagen, representa el servicio, que ofrece los dispositivos. Cada servicio tiene un conjunto lógicamente coherente de acciones que están estrechamente asociados con las variables de estado. Esas variables, marcadas en la imagen por medio de símbolos "var1" y "Var2" representan las características del servicio. Estas características (es decir variables) aceptan

ciertos valores que definen claramente las condiciones del servicio y el dispositivo, a que pertenece el servicio. Si el valor de la variable de estado ha cambiado, entonces esto significa, que el estado del dispositivo también ha cambiado, el cual a su vez indicara este cambio mediante la notificación del evento y recibe la solicitud del punto de control, utilizando un objeto de devolución de llamada, de esta manera se realiza el seguimiento del estado actual del dispositivo. El servicio del dispositivo (que es objeto de servicio) ofrece un conjunto de acciones posibles para cumplir con el servicio. Las acciones son utilizados por la aplicación de punto de control para recuperar información sobre el estado del dispositivo y ordenar los cambios en su estado o realizar ciertas tareas, es decir, permite el control del dispositivo, la aplicación de punto de control, afecta a través de acciones las variables de estado, provocando algunas reacciones específicas del dispositivo, por ejemplo, reduciendo la intensidad de la fuente de luz. Cada acción puede estar asociada con una o más variables de estado de la siguiente manera, que la acción no se atribuya a ninguna variable. En la figura 30, tenemos un ejemplo en acción, "Acción 1" provoca un cambio de la variable "var1" y la "Accion3" cambios en la variable "Var2", mientras que la acción "Acción 2" provoca la reacción de ambas variables "var1" y "Var2".

3.2.2. Interfaces Windows XP relacionados con el objeto Gestor dispositivos

La API de puntos de control es un conjunto de varias interfaces COM, que se pueden dividir según el tipo de objeto con el que estén asociados. Esta selección se refiere a los tres objetos básicos discutidos antes. El primer grupo es la referencia a las interfaces " Objeto Buscador de Dispositivos", que como su nombre indica, se ocupa de la detección de dispositivos, o para ser más precisos, aplica las tareas asociadas con la vigilancia de los procesos de conexión y desconexión de dispositivos en la red.

• IUPnPDeviceFinder
• IUPnPDeviceFinderCallback

IUPnPDeviceFinder, es una de las principales interfaces del API de punto de control y se utiliza para crear un objeto Buscador de dispositivo. Por medio de ella se puede encontrar los dispositivos a través del método sincrónico o asincrónico con el objeto de devolución de llamada. En muchos casos (como las aplicaciones GUI) el método asíncrono será una mejor opción porque no bloquea la interfaz de usuario del programa.
El método asincrónico consta de cuatro pasos principales, El primer paso es crear un objeto de devolución de llamada, que recibirá los resultados de búsqueda. En el siguiente paso llamamos la función CreateAsyncFind pasándole el puntero al objeto de devolución de llamada. El siguiente paso es comenzar la búsqueda real utilizando la función StartAsyncFind. El último paso es devolver el resultado de la búsqueda de entorno UPnP en la devolución de llamada.

Para utilizar el método asincrónico, se necesita un objeto de devolución de llamada. Para crear una instancia de este objeto, primero se debe preparar la clase que implementa la interfaz IUPnPDeviceFinderCallback que permite recibir los resultados de la búsqueda asincrónica del frame UPnP.

3.2.3. Interfaces de Windows XP relacionados con el objeto Dispositivo.

El siguiente grupo de interfaces está relacionado con objeto de dispositivo:

• IUPnPDevice,
• IUPnPDevices,
• IUPnPDeviceDocumentAccess,

• IUPnPDescriptionDocument,
• IUPnPDescriptionDocumentCallback.

El más importante de este grupo es la interfaz IUPnPDevice a través de la cual se puede acceder al objeto Dispositivos. No es posible crear esta interfaz, ya que es proporcionada por el entorno UPnP como resultado de la búsqueda de productos que circulan en la red. Mediante el uso de las funciones de esta interfaz, se puede leer una gran cantidad de información sobre el dispositivo (propiedades) como su propio nombre, el nombre del modelo, el nombre del fabricante o la dirección, que se implementara en un navegador para tener acceso a las opciones de configuración de dispositivos.

Por otra parte, las funciones de la interfaz IUPnPDevice permiten el examen de la estructura del dispositivo que en su estructura lógica interna:

• get_HasChildren,
• get_Children,
• get_Services.

Con la función get_HasChildren podemos saber si el producto contiene dispositivos miembro (dispositivos secundarios) en su interior. Si la función get_HasChildren devuelve el valor lógico "verdadero", entonces se puede obtener un conjunto de dispositivos miembro llamando a get_Children. Una colección de dispositivos de usuario es devuelto en forma de interfaz IUPnPDevices. Tenga en cuenta que la estructura lógica de un dispositivo es de árbol, por lo tanto, cada dispositivo puede incluir miembros de una colección de sus dispositivos propios. Para hacer frente a la identificación de la estructura de árbol a través de la recursividad, se llama a la función get_Children como función auxiliar recurrente. La función get_Services nos devuelve un conjunto de servicios del dispositivo, como objetos de servicio, en forma de interfaz IUPnPServices.

3.2.4. Interfaces de Windows XP relacionados con el objeto Servicio.

El último grupo de interfaces pertenecientes al API de Punto de Control se relaciona con el objeto Servicio y son:

- IUPnPService,
- IUPnPServices,
- IUPnPServiceCallback.

El acceso al objeto Servicio sólo puede hacerse a través de un objeto Dispositivo. La interfaz principal del objeto Servicio es IUPnPService. Para obtener un conjunto de servicios de determinado producto, usted debe llamar a la función get_Services de la interfaz IUPnPDevice. Esta función devuelve el objeto Servicios al que tenemos acceso a través de la interfaz IUPnPServices.

IUPnPService permite a una aplicación recuperar información de estado y ejecutar acciones para un servicio.

IUPnPServiceCallback devuelve una solicitud para notificar al entorno UPnP en caso de ocurrir eventos mayores.

IUPnPServices Enumera una serie de servicios.

En el proyecto se implementa en primer lugar, la consulta a la interfaz IUPnPDevice:: Servicios, propiedad de la interfaz que se pasa a la función. Esto devuelve una colección de servicios que utilizan la interfaz IUPnPServices. Para obtener los objetos individuales de servicio, se utiliza el método descrito

anteriormente y se especifca el ID del servicio solicitado. Para recorrer la colección de forma secuencial, se utiliza el IEnumVariant::Reset, IEnumVariant.
Cada servicio, así como el mismo dispositivo, se describen en el fichero XML - Documento de designación, dedicado al servicio dado. La estructura de este documento también debe ser compatible con el estándar UPnP dado por la organización del Foro UPnP. El documento de servicio incluye datos de las acciones, sus nombres y argumentos y detalles sobre las variables de estado asociados a las acciones. La estructura del documento de servicio se compone de dos listas: acciones y variables de estado.
Para cada acción se especifica su nombre y la lista de argumentos. Si la acción tiene argumentos, a continuación, se dan sus nombres, dirección de cruce (entrada / salida) y el nombre de variable de estado relacionados. Para las variables de estado se definen datos como: nombre, tipo, los valores permitidos, el valor mínimo y máximo, el paso y el valor predeterminado.

El documento con los archivos de servicios se puede descargar desde el dispositivo. Pero, ¿dónde encontrar las direcciones URL de estos documentos? La URL se guarda en forma relativa (por lo general pero no necesariamente), en el documento de un dispositivo, en parte relacionado con el servicio dado, en la etiqueta llamada <SCPDURL>. La parte base de la URL, de conformidad con la norma, se deben colocar en el documento del dispositivo, en la etiqueta llamada <URLBase>.

Así, los pasos para la obtención de datos acerca de las acciones de cada dispositivo son los siguientes:

• Leer la dirección del documento del dispositivo de la manera antes indicada en la descripción de la interfaz de IUPnPDeviceDocumentAccess.
• Transferir los archivos de descripcion del dispositivo incluidos en el documento.

• Examinar el contenido del documento del dispositivo, leer la dirección del documento del servicio de la etiqueta SCPDURL (si la URL es relativa a continuación, añadir a la URL base lectura de la etiqueta URLBase).
• Descargar el fichero del servicio de descripción de documentos.
• Examinar el contenido del documento Servicio, es decir leer los datos de que nos interesan a nosotros.

3.3. ARQUITECTURA DE CLASES

En este apartado se describen el conjunto de paquetes y clases que componen el proyecto de desarrollo del Gestor de dispositivos UPnP. También, por su importancia, se mencionan algunos métodos definidos en ellas.
En la siguiente figura se muestra el diagrama de clases del Gestor de Dispositivos. Se recomienda consultar el “ANEXO II” sobre Diagramas UML para su mejor interpretación.
Se han implementado unas cuantas clases que definen los objetos más importantes de la arquitectura UPnP, así como un conjunto de funciones de utilidad, las interfaces y estructuras que son útiles en el cumplimiento de las funciones básicas de la aplicación del Gestor UPnP.
La librería principal consiste en un archivo de encabezado GestorDispositivos.h y GestorDispositivos.cpp. Para procesar los archivos XML se utiliza el software CMarkup, bajo las condiciones de "Licencia de Evaluación CMarkup”. Para compilar la librería se debe agregar al proyecto la aplicación y los archivos markup.h markup.cpp

La librería GestorDispositivos contiene cuatro clases principales que definen los cuatro objetos más importantes de la arquitectura UPnP.

• **Buscador** - se refiere al objeto de dispositivos del Buscador.

• **Dispositivo** - corresponde al dispositivo UPnP.

• **Servicio** - corresponde al servicio objeto de dispositivo UPnP.

• **Acción** - corresponde a la acción de servicios.

Además, contiene dos clases de ayuda:

• DevFinderCallback - implementa la interfaz IUPnPDeviceFinderCallback.

• SrvEventCallback - implementa la interfaz IUPnPServiceCallback.

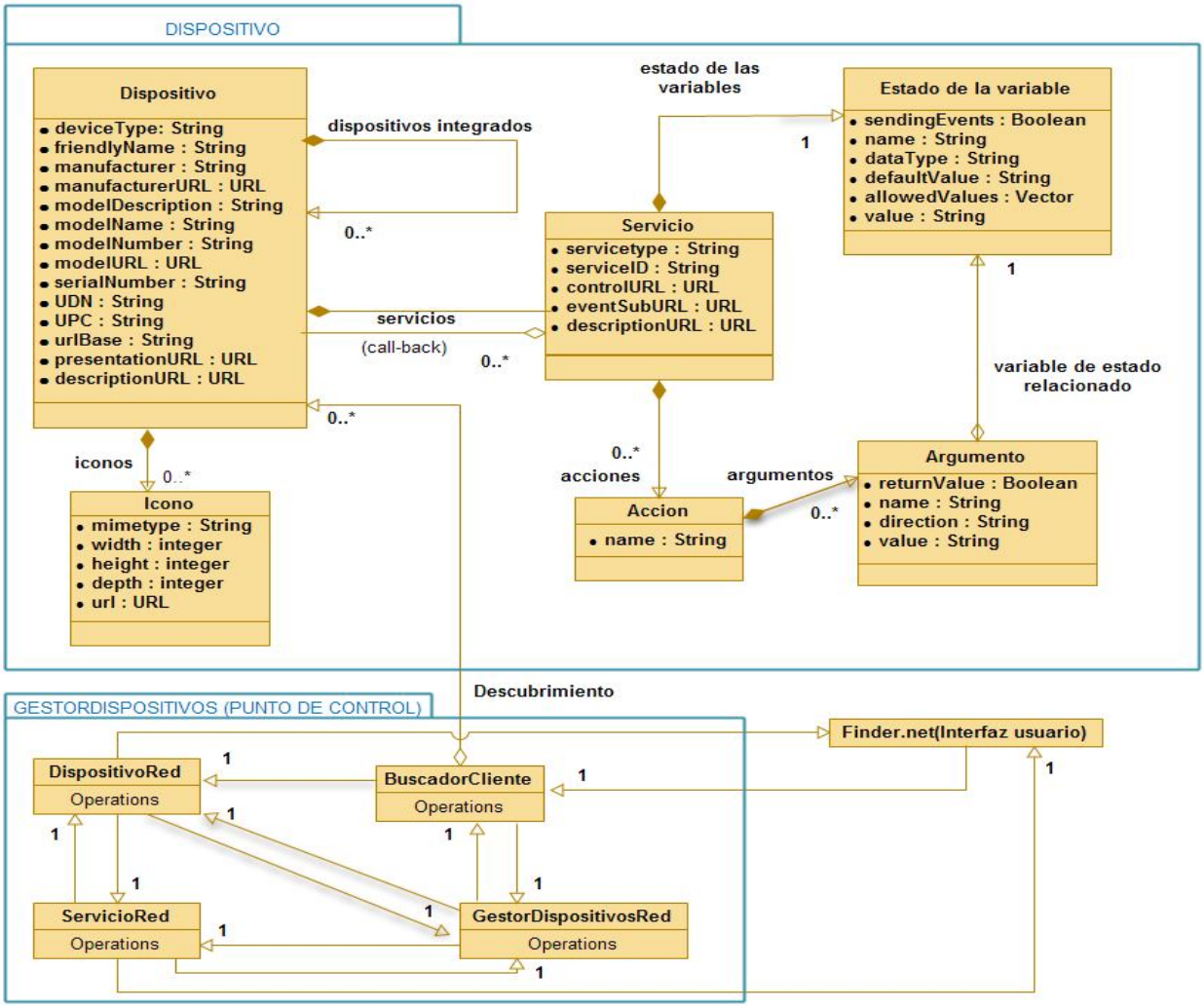


Figura 31. Diagrama de clases del Gestor UPnP

FUENTE: Autores

3.3.1. Clase GestorDispositivos

La clase GestorDispositivos se utiliza para administrar el objeto dispositivo. Automáticamente crea y destruye el objeto buscador y controla el proceso de descubrimiento. Es la única clase que puede crear objetos por sí misma. El resto de objetos son creados por el frame automáticamente y se pueden acceder a través de los métodos de otros objetos.

Usar la clase GestorDispositivos es bastante simple. Después de crear el objeto, simplemente se llama a la función init que sobrecarga y adecua el próximo inicio de búsqueda. Las funciones iniciar y detener, respectivamente, cumplen con ejecutar y detener el proceso de descubrimiento de dispositivos. Esta clase permite administrar una colección interna de los dispositivos detectados (objetos de la clase de dispositivo). También puede almacenar una colección de dispositivos descubiertos, en una clase propia implementada a través de la interfaz IFinderCallbackClient.

GestorDispositivos administra una colección que coopera con la implementación de la interfaz de cliente IFinderManagerClient, para que el cliente sea notificado sobre los eventos de inicio y fin del proceso de descubrimiento, así como la adición y la eliminación de objetos de la colección objetos. GestorDispositivos puede pasar estas notificaciones, para un cliente externo, que implementa la interfaz IServiceCallbackClient. GestorDispositivos utiliza la clase DevFinderCallback para recibir del entorno UPnP (objeto Buscador de dispositivos) los resultados de la búsqueda de dispositivos en la red.

3.3.2. Clase Dispositivo

Esta clase es uno de los elementos básicos de la arquitectura UPnP, el Dispositivo UPnP. Esta clase refleja una estructura de árbol del dispositivo y su funcionalidad de acuerdo con el estándar de arquitectura de dispositivos de la organización Foro UPnP. La clase dispositivo contiene tres colecciones:

- DeviceList: Colección de los objetos del tipo de dispositivo que representa los dispositivos miembros.
- ServiceList: Colección de objetos del tipo de servicio que representan los servicios de los dispositivos miembros.
- IconList: Colección de las estructuras de tipo IconParam que contiene los parámetros de los iconos.

Además, el dispositivo contiene información como el nombre del dispositivo y los datos relativos al documento de descripción de la estructura DocAccessData. En el objeto de servicio se almacena una colección de objetos de acciones disponibles en el servicio.
La clase Dispositivo tiene un conjunto de funciones para la manipulación de las colecciones mencionadas anteriormente: añadir, eliminar y enumerar elementos. Tiene también la función EnumerateDevices, que enumera los objetos de forma recursiva en la estructura de árbol de dispositivos miembro. La enumeración se puede iniciar desde cualquier nodo de árbol y se continúa profundizando la estructura.

3.3.3. Clase Servicio

La clase Servicio representa el objeto de servicio de los dispositivos UPnP.
La clase Servicio contiene una lista de objetos acciones (ActionList) pertenecientes al servicio que representa, y también almacena la dirección y el

contenido de descripción de los documentos de servicio, donde se almacenan todos los datos necesarios para realizar la acción.

Cada objeto servicio crea su propia instancia de la clase SrvEventCallback (implementación de la interfaz IUPnPServiceCallback) para recibir notificaciones de eventos de los servicios. Las notificaciones recibidas se envían hacia el cliente del objeto de devolución de llamada, es decir, con el objeto de clase que implementa la interfaz IServiceCallbackClient. El objeto de devolución de llamada (SrvEventCallback), que recibió una notificación de entorno UPnP sobre el evento en el servicio, recibe al mismo tiempo, un puntero al objeto de servicio (* IUPnPService).

3.3.4. Clase Acción

Esta clase contiene información sobre la entrada de argumentos y la función Invoke que invoca las medidas adecuadas para la ejecución de los servicio de los dispositivos.

3.4. DIAGRAMA DE SECUENCIAS

A fin de poder comprender mejor el funcionamiento de la aplicación, se han elaborado los siguientes diagramas UML de secuencia de los principales procesos de la misma. Se recomienda consultar el “ANEXO II: Diagramas UML” donde se describe la simbología utilizada.

Estos diagramas muestran de forma esquemática la secuencia de invocación de métodos que realiza el sistema para la implementación de una funcionalidad.

Los principales procesos, por su importancia en el diseño de la aplicación, son:

1. Anuncio.
2. Descubrimiento.
3. Descripción de servicios.
4. Control.
5. Notificación de eventos.
6. Presentación.

Cuando un dispositivo se une a una red, se anuncia a la red con cierta frecuencia, a la cual llamamos anuncio. Cuando un punto de control se une a una red, busca los dispositivos que puede controlar, a este paso se le llama descubrimiento. El protocolo SSDP incluye tanto Publicidad (anuncios) como Descubrimiento.

Aquí se incluye algunas definiciones de las siguientes clases:

HTTPServidor: Un servidor HTTP se ejecuta en el punto de control para manejar la Publicidad de los dispositivos.

GestorDispositivosRed: Procesa los datos facilitados por el HTTPServidor; además controla las respuestas de los dispositivos cuando el punto de control realiza los procesos de descubrimiento.

Finder.net (interfaz del usuario): Inicia el descubrimiento por parte del Gestor UPnP.

Advertiser: Inicia el anuncio por parte del dipositivo.

SSDPSolver: Son los procesos de los datos facilitados por HTTPServidor/UDP.

SOAPSolver: Corre en el dispositivo para manejar los comandos de control expedido por el punto de control.

El componente GENA implementa el proceso de concurso completo e Incluye las clases:

SubsSolver: Se ejecuta en el dispositivo para manejar las solicitudes de subcripción desde el punto de control.

EventSolver: se ejecuta en el punto de control para controlar los eventos enviados por el dispositivo.

El formato de transferencia de datos en el servidor web esta conforme al protocolo HTTP con el propósito de cumplir con el requisito de la SSDP, GENA y SOAP. En la capa del UDP y el TCP hay dos clases que son esenciales: HTTPServidor/UDP (UDPListener) y TCPListener.

HTTPServidor/UDP es un servidor HTTP ejecutándose encima de UDP para manejar el proceso de descubrimiento de servicios en el SSDP.
TCPListener es un servidor HTTP ejecutándose encima de TCP para hacer frente a las exigencias de SOAP y Gena.

3.4.1. Anuncio

En el siguiente diagrama UML se observa el proceso de anuncio del dispositivo, el primer paso a llevar a cabo es ingresar a la red al dispositivo UPnP que tenemos a disposición, para esto la secuencia se comienza con la invocación del método Init() de arranque de la aplicación, entonces continuando con la ejecución del método Init() , la siguiente instrucción es la invocación al método con el mismo nombre Init() donde se crean la instancia del Advertiser que inicia el envió de

mensajes de anuncio periódicos por parte del dispositivo a la red, tal y como lo define el estándar UPnP.

Como consecuencia, se desencadena todo el proceso de inicialización y anuncio del dispositivo. Este proceso implica la creación de la lista de servidores HTTP y sockets para la búsqueda SSDP.

El primer paso antes de cualquier interacción entre dispositivos, es la asignación de una dirección IP, fase que denominamos de direccionamiento. UPnP usa un mecanismo de direccionamiento hibrido basado en el protocolo Dynamic Host Configuration Protocol (DHCP) como mecanismo centralizado y en Dynamic Configuration of IPv4Local-link Addresses5 como mecanismo distribuido.

Un dispositivo envía un mensaje DHCP Discover al servidor. Si un servidor DHCP recibe este mensaje, entonces responde con un mensaje DHCP Offer dirigido hacia el dispositivo. En este mensaje, el servidor incluye una propuesta de dirección IP. Si el dispositivo acepta esta propuesta, entonces este envía un mensaje DHCP Request al servidor. Finalmente, el servidor reserva esa dirección IP para que otros clientes no la puedan usar y envía un mensaje DHCP Ack al dispositivo. Si, después de enviar el mensaje DHCP Discover, el dispositivo no recibe respuesta alguna de un servidor, entonces procede a ejecutar el mecanismo distribuido de asignación de direcciones.

Una vez que el dispositivo ya tiene su dirección IP asignada entonces notifica (NOTIFY) al servidor de que ya está puesto en marcha; el servidor recibe la información y la envía al punto de control en donde es recibida por el GestorDispositivosRed que es la clase del punto de control que se encarga de recibir esta información y adiciona el dispositivo (OnAddDevice) a la recogida realizada y luego se actualiza la información obtenida (GetCollection) cuya información es maneja por la librería GestorDispositivos.

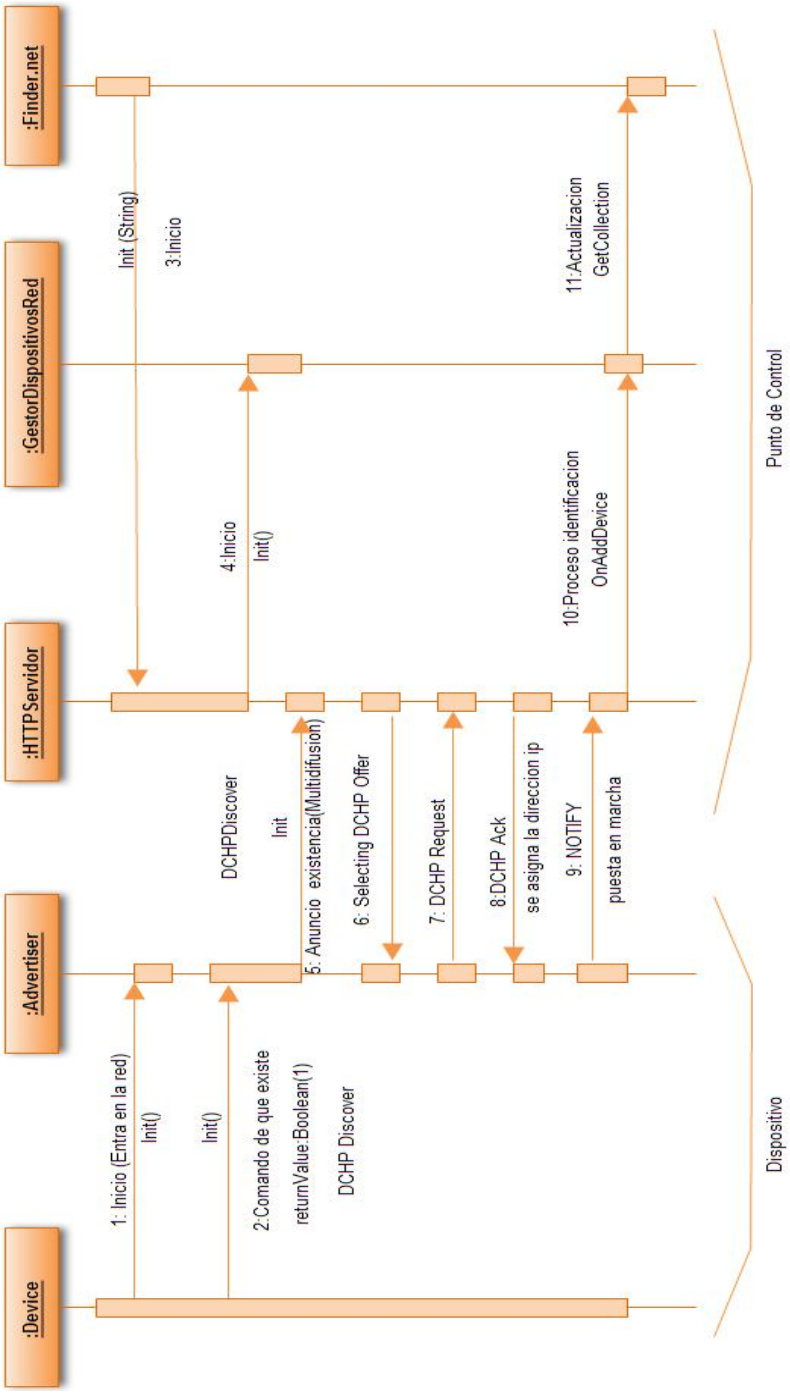


Figura 32. Diagrama de Secuencia para el Anuncio.

3.4.2. Descubrimiento

La fase de descubrimiento se basa en el protocolo Simple Service Discovery Protocol (SSDP). Este protocolo permite que los dispositivos subordinados puedan ofrecer servicios y que puntos de control sean capaces de encontrar estos servicios para, posteriormente, interaccionar con ellos. De todas formas, a este nivel no hablaremos de dispositivos subordinados o puntos de control; esto es debido a que SSDP es un protocolo completamente distribuido o Peer to Peer en el que cualquier dispositivo participante puede ser, al mismo tiempo, servidor y cliente. Usaremos la palabra nodo o host para referirnos a un dispositivo que ejecuta el protocolo SSDP, ya sea proveedor o usuario de servicios.

(Dispositivo). Cuando un nodo se pone en funcionamiento, después ejecutar la fase de direccionamiento, pasa a la fase de descubrimiento. Así, el primer paso consiste en anunciar los servicios que el propio nodo ofrece; en el caso de UPnP, serán los dispositivos los que realizarían esta tarea. La publicación (o anuncio) de los servicios se lleva a cabo enviando un mensaje *NOTIFY*, que son un tipo de mensaje HTTP definido dentro de la arquitectura de notificación GENA cuyo mensaje es enviado al servidor (HTTPServidor). Como hay varias clases de mensajes *NOTIFY* definidos, es necesario aclarar que este pertenece al protocolo SSDP. Estos mensajes se encapsulan en paquetes UDP dirigidos a una dirección IP y puerto (239.255.255.250:1900) definidos en el estándar.

Por otro lado se inicia (Init()) la interfaz del punto de control que da inicio al BuscadorCliente que se encarga de buscar los dispositivos que hayan disponibles en la red, y este a su vez reinicia al GestorDispositivosRed para controlar las respuestas de los dispositivos. Se inicia el BuscadorCliente (OnStartFind) que se comunica con el servidor disponible en la red.

(BuscadorCliente al HTTP Servidor al SSDPSolver). Ahora usaremos el método de HTTP denominado M-SEARCH. Este tipo de mensaje HTTP se encapsula dentro de un paquete UDP multicast y se envía a la misma dirección IP multicast y puerto previamente definidos para los mensajes ssdp:alive. Esta información pasa del BuscadorCliente al servidor HTTPServidor y luego pasa al SSDPSolver.

(SSDPSolver al Dispositivo).Cuando un nodo recibe un mensaje del tipo *ssdp:discover*, comprueba si se ajusta a las condiciones de la busqueda que expresa el campo *ST* de la cabecera. Si es así, entonces se responde con un mensaje *HTTP 200 OK* encapsulado en un paquete UDP *unicast* dirigido hacia el nodo que originalmente envió el mensaje *ssdp:discover*.

(Dispositivo). Al igual que en el mensaje *ssdp:alive*, el campo *LOCATION* de la cabecera almacena la URL que apunta al servidor web del nodo; en concreto, indica la localización del documento de descripción de dispositivo. Estos mensajes incluyen también información útil como el tipo de dispositivo o servicio que se ofrece.
(Del SSDPSolver al BuscadorCliente). Estos mensajes se encapsulan en paquetes UDP dirigidos a una dirección IP del punto de control que van del Dispositivo al SSDPSolver y luego al BuscadorCliente.

Del BuscadorCliente la información pasa al GestorDispositivosRed (AddDeviceArgs) que controla las respuestas de los dispositivos y luego la información es actualizada en el cache para ser mostrada a través de la interfaz del punto de control que es el finder.net.

Finalmente, existe una función que permite a un nodo indicar que va a ser puesto fuera de servicio. Se procede de la misma manera que cuando un nodo quiere anunciar su presencia mediante mensajes *ssdp:alive*. En este caso, se modifica el nombre de los mensajes y se incluye la cadena de caracteres *ssdp:byebye* en el

campo *NTS*. Se envía exactamente el mismo número de mensajes *ssdp:byebye* que mensajes *ssdp:alive* se enviaron para anunciar la presencia del dispositivo.

Por tanto, cuando un *Control Point* recibe un mensaje de anuncio del dispositivo, ya puede obtener la información de su descripción a partir de la URL que se especifica en el mensaje SSDP recibido. Si un dispositivo no es descubierto tras este mensaje de anuncio, normalmente porque el *Control Point* haya arrancado posteriormente, recibirá un mensaje SSDPSearch de éste y procederá a enviarle una respuesta SSDP con la URL de su descriptor.

El servidor HTTP, por medio de un *socket* HTTP, será el que reciba todos los mensajes de solicitud que envíe el *Control Point* utilizando este protocolo. Entre estos mensajes están la solicitud del descriptor del dispositivo, las solicitudes de subscripción y las invocaciones de las acciones.

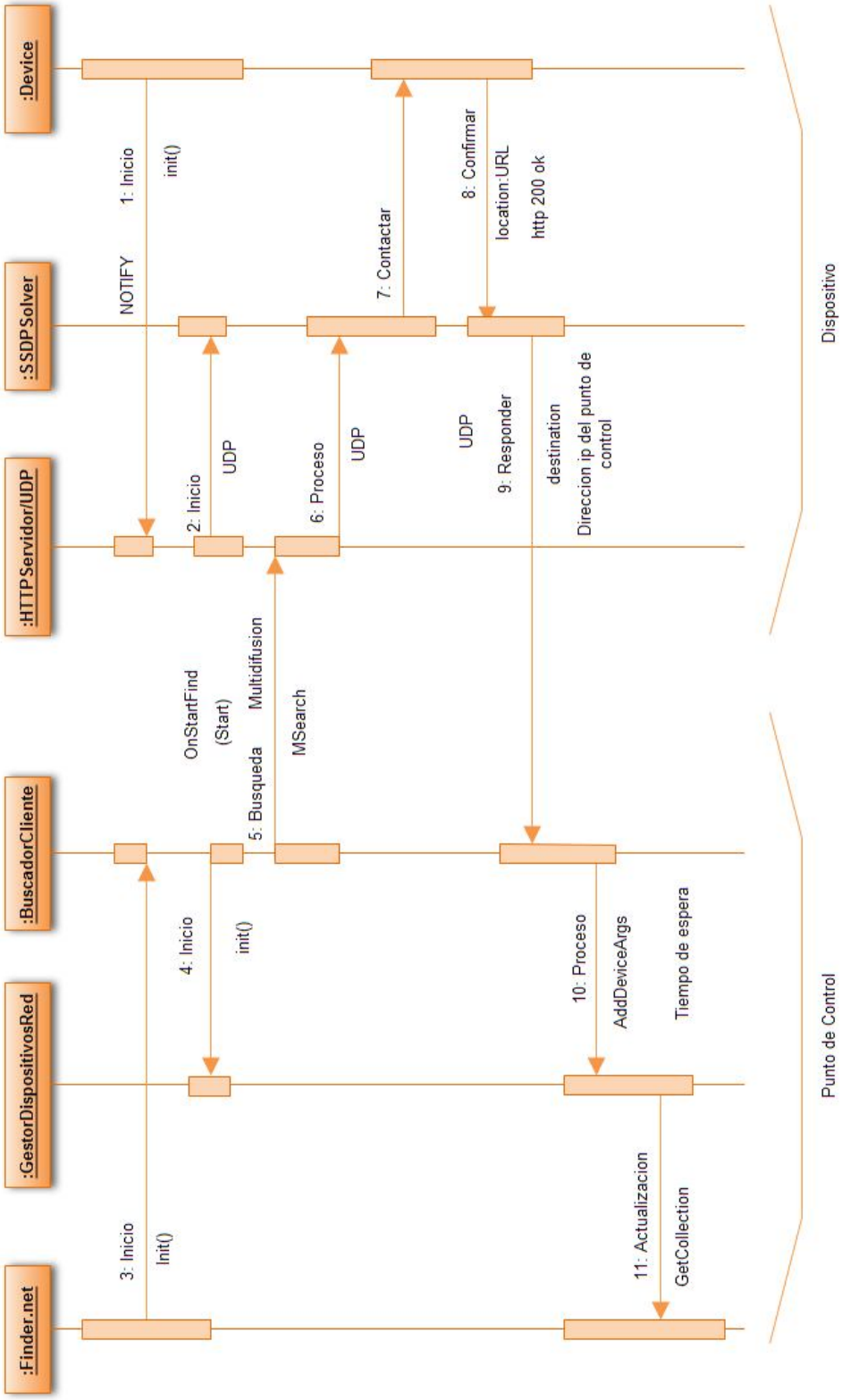


Figura 33. Diagrama de secuencia para el Descubrimiento.

3.4.3. Descripción de servicios

Después de la fase de descubrimiento, los dispositivos cuentan con información sobre donde se encuentran los documentos de descripción de dispositivo y servicio, necesarios para poder usar los servicios. La fase de descripción consiste en trasladar estas descripciones de unos equipos a otros y por supuesto se basa en la fase de descubrimiento es decir, ahora en esta fase solo le envía la información al punto de control, información que en usuario desea conocer. Se trata de documentos XML, donde están escritos todos los datos sobre el dispositivo y su estructura. El documento requiere el elemento dispositivo y su formato y la estructura se especifica en la norma de la organización del Foro UPnP.

El documento en forma de archivo XML es preparado por el fabricante del dispositivo y se coloca en el dispositivo; con lo que dichos documentos XML pueden ser transferidos entre equipos usando el protocolo HTTP sobre TCP.

Vamos a ver cómo obtener acceso al documento de descripción; para esto usamos la clase DispositivoRed que se usa en el punto de control. Sólo tiene una función GetDocumentURL, como pueden suponer, devuelve la dirección URL del documento; y también usamos la otra función GetDocumentContent que devuelve la descripción del documento.

Los puntos de control envían mensajes HTTP GET a un dispositivo subordinado indicando la localización del documento en cuestión en forma de URL. A su vez, el dispositivo subordinado responde enviando un mensaje HTTP 200 OK incluyendo el documento en el cuerpo del mensaje.

No se muestran paquetes TCP, sino únicamente mensajes HTTP. En primer lugar se pide la descripción del dispositivo raíz. Esta, a su vez, contiene las direcciones URL de las descripciones de otros dispositivos empotrados y servicios.

A continuación se presenta una grafica que representa el proceso anteriormente tratado y que se espera, aclare un poco más su funcionamiento.

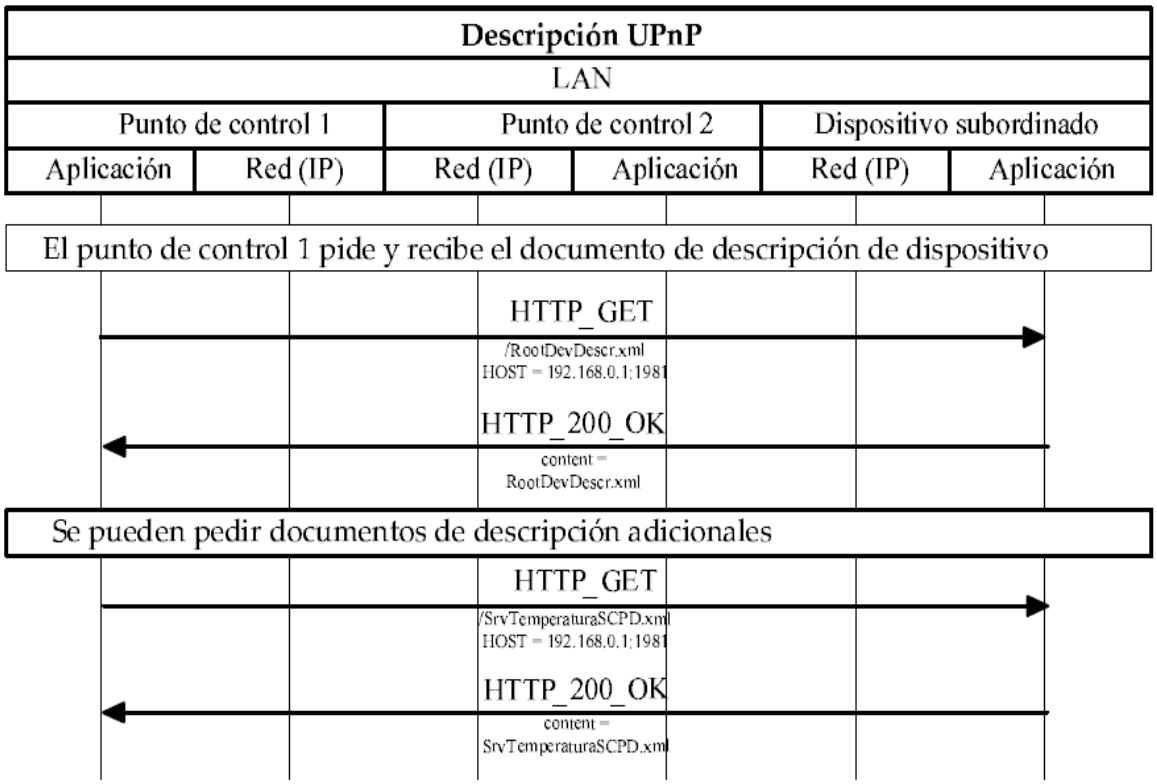


Figura 34. Intercambio de mensajes HTTP entre dispositivos y puntos de control

FUENTE:
http://webpersonal.uma.es/~ECASILARI/Docencia/Memorias_Presentaciones_PFC/28PFC.pdf

3.4.4. Control

Una vez que un punto de control ha obtenido todos los documentos, entonces puede comenzar la siguiente fase, que consiste en el uso de los servicios. La fase de control se basa en el protocolo simple Object Access protocol (SOAP). Se trata de un mecanismo RPC[4] que permite que los puntos de control puedan invocar acciones y consultar el valor de variables de estado en dispositivos remotos. Estas acciones y variables corresponden a aquellas descritas en los documentos de descripción ya obtenidos. SOAP consigue todo esto sin necesidad de instalar controladores en ninguno de los dispositivos.

Se definen dos tipos de operaciones:

3.4.4.1. Invocación de acciones

Dentro del punto de control tenemos dos clases que son la clase dispositivo(CDevice) y la clase servicio (CService); para tener acceso a los servicios tenemos que realizar el llamado de la función GetServices, luego desde la clase servicios invocamos la acción a través de la función InvokeAction luego ya abarcando esto, desde el punto de control envía un mensaje HTTP del tipo POST a la dirección indicada por la etiqueta controlURL del documento de descripción del dispositivo donde reside el servicio que se desea controlar. La cabecera de este mensaje debe especificar la acción a invocar. Si es necesario, las variables de entrada se incluyen como un documento XML dentro del cuerpo del mensaje. El dispositivo subordinado, a su vez, procesa la solicitud y responde con un mensaje HTTP 200 OK, desde el punto de control en la case servicio se obtiene las acciones a traves de la función GetActions. Su cuerpo es un documento XML que incluye cualquier variable de salida fruto de la invocación de la acción.

[4] RPC: Llamadas a procedimientos de forma remota.

3.4.4.2. Consulta de variables

En lugar de especificar una acción a realizar, el punto de control incluye una lista de variables de estado cuyo valor se desea conocer, para esto utiliza la función QueryStateVariable desde la clase servicio. El dispositivo subordinado debe responder con un mensaje HTTP 200 OK cuyo cuerpo contenga un documento XML donde se detallen los valores actuales de las variables solicitadas, el punto de control en la clase servicio utiliza la función StateVariables para conocer el estados de las variables.

En el caso de que alguna acción o variable de estado no esté disponible o no exista, los dispositivos subordinados devolverán un mensaje de error con un código explicativo de la razón del error.

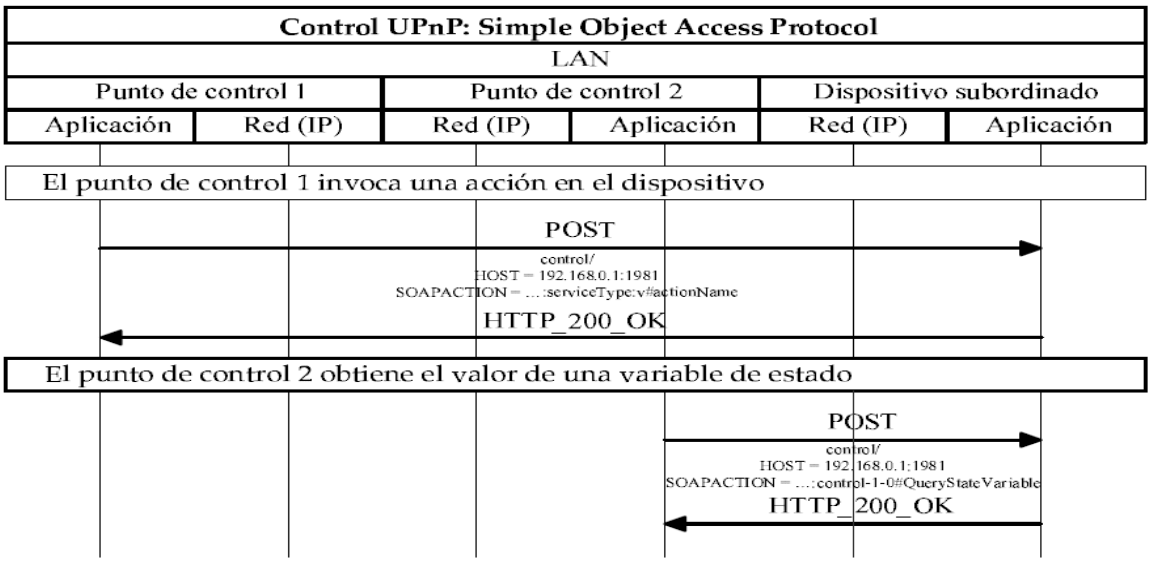


Figura 35. Consulta variables de estado

FUENTE:
http://webpersonal.uma.es/~ECASILARI/Docencia/Memorias_Presentaciones_PFC/28PFC.pdf

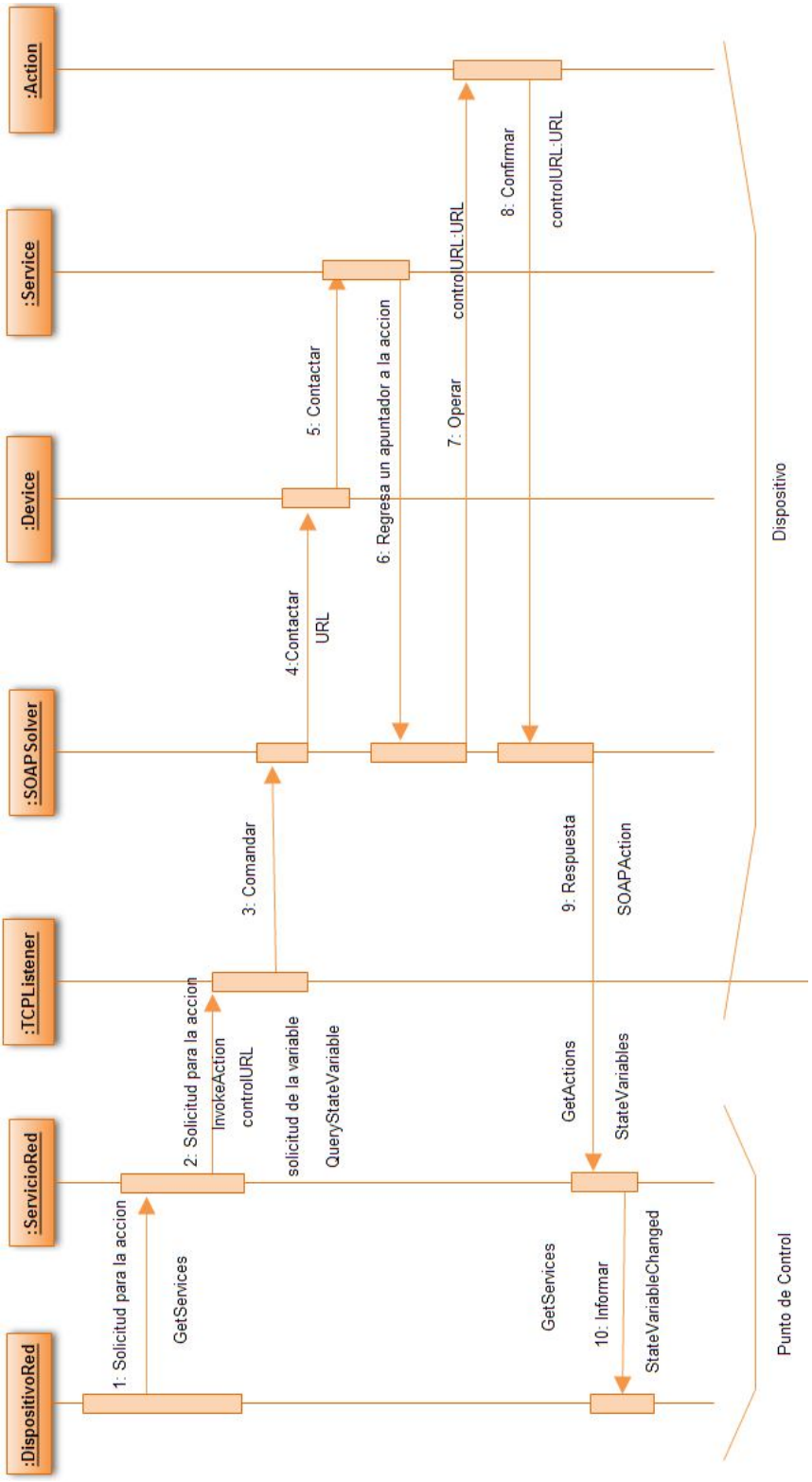


Figura 36. Diagrama de secuencia para el Control.

3.4.4.3. Notificación de eventos

La notificación de eventos permite que los dispositivos subordinados mantengan informados a los puntos de control de cualquier cambio que ocurra en el estado de los mismos. La arquitectura de notificación de eventos utilizada se denomina Generic Event Notification Architecture (GENA), basada en HTTP sobre TCP.

En la figura 37 se muestra un ejemplo de un proceso completo de notificación de eventos. Cuando un punto de control desea ser informado de cualquier cambio que suceda dentro de un dispositivo, entonces debe suscribirse. En el diagrama UML tenemos en cuenta que ya se encuentra funcionando la clase servicio que se comunica con el servidor y este a su vez a iniciado al EventSolver (encargado de controlar los mensajes enviados por los dispositivos); la clase servicio maneja la función QueryStateVariable para conocer el estado de la variable con la cual se esté trabajando, entonces para realizar este paso automáticamente gracias a las interfaces de Microsoft y de acuerdo a los estándares del protocolo UPnP, la suscripción se realiza enviando un mensaje HTTP del tipo SUBSCRIBE a la dirección de suscripción indicada por la etiqueta eventSubURL del documento de descripción del dispositivo. El punto de control debe incluir información adicional sobre la localización de su servidor web en el campo CALLBACK de la cabecera. A este mensaje, el dispositivo subordinado debe responder con un mensaje HTTP 200 OK que contiene dos datos importantes

1. El identificador de la suscripción: Usado por el punto de control para renovar la suscripción más adelante. También lo usan los dispositivos subordinados para enviar notificaciones de cambio de estado a los puntos de control.

2. La duración de la suscripción: El periodo durante el cual la suscripción es válida. Antes de finalizar este periodo, y si se desea seguir recibiendo

notificaciones, el punto de control debe renovarla enviando otro mensaje de suscripción.

El mensaje HTTP 200 OK llega al servidor del punto de control (TCPListener) y este a su vez lo envía al EventSolver que se encarga de controlar los eventos enviados por el dispositivo; la clase servicio maneja esta información a través de la función SrvEventCallback que se encarga de recibir notificaciones de los eventos de los servicios.

Cada vez que una variable cambia su estado, los dispositivos subordinados envían un mensaje HTTP NOTIFY a los puntos de control que tienen suscripciones validas.

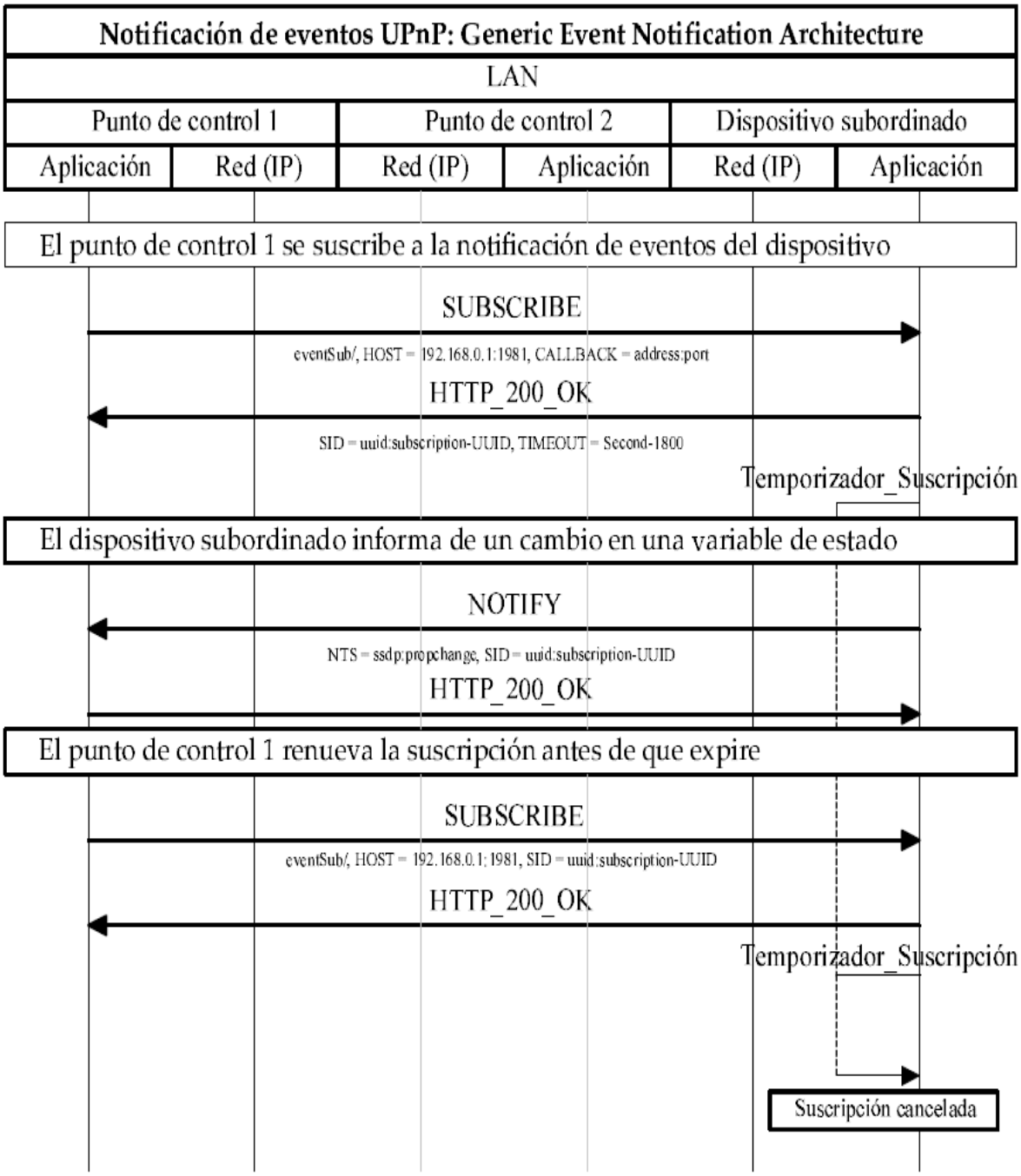


Figura 37. Esquema del proceso de Notificación de Eventos

FUENTE:
http://webpersonal.uma.es/~ECASILARI/Docencia/Memorias_Presentaciones_PFC/28PFC.pdf

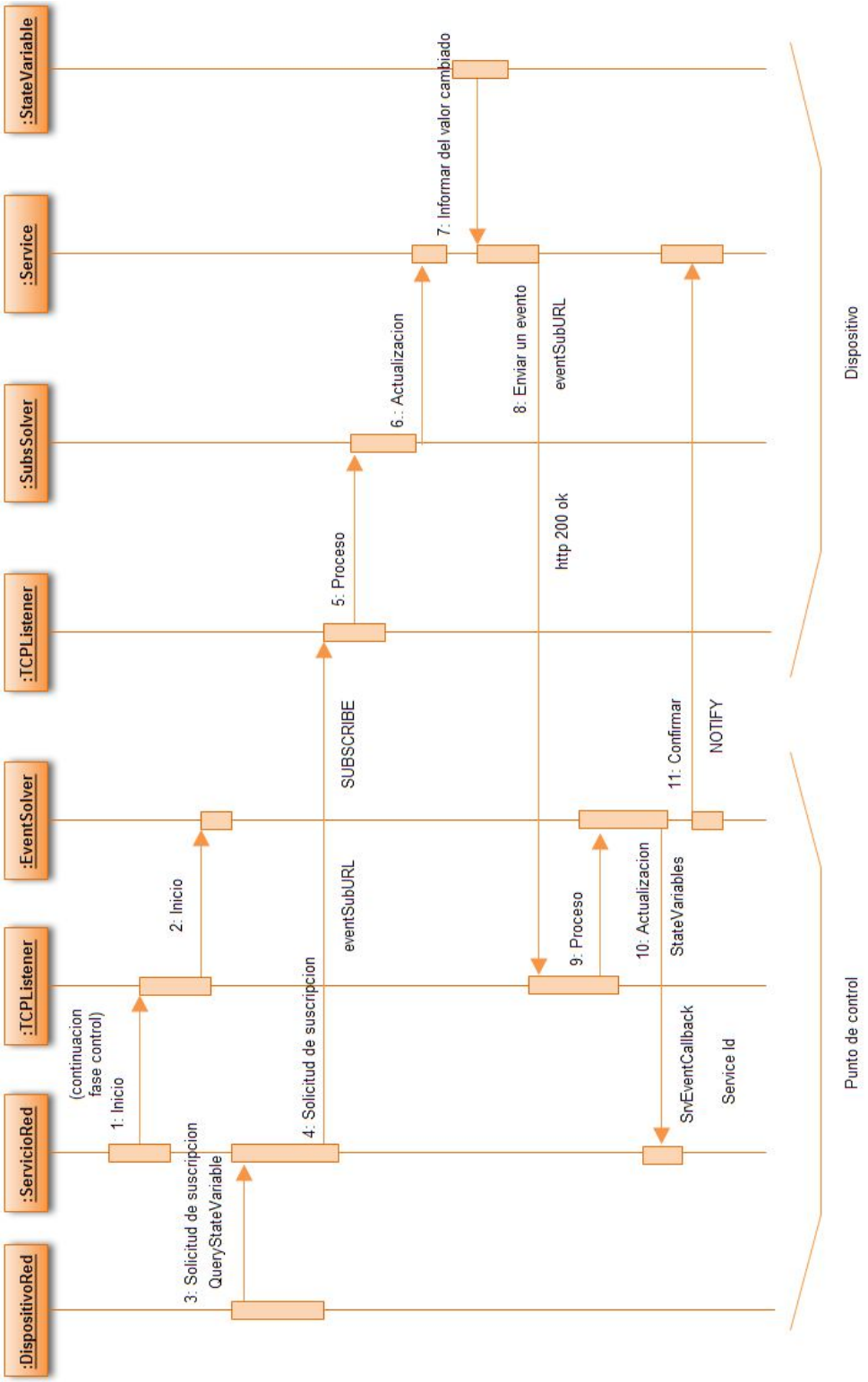


Figura 38. Diagrama de secuencia para la Notificación de Eventos

3.4.5. Presentación

La presentación es una fase complementaria al control y a la notificación de eventos. Se ofrece una alternativa de control de los dispositivos fácil de utilizar para el usuario final. Si un dispositivo posee una URL para presentación (descrita por la etiqueta presentationURL del documento de descripcion de dispositivo), entonces el usuario puede acceder a través de ella a una página web para el control del dispositivo. El diseño y características de esa web esta fuera del alcance de UPnP. Su complejidad determinara el grado de control que el usuario pueda ejercer sobre el dispositivo.

4. PRUEBAS

En este capítulo se recogen las pruebas realizadas a la aplicación con el objeto de valorar la respuesta del sistema en un escenario real de ejecución y presentar datos de su eficiencia.

4.1. Introducción

Los objetivos que se persiguen con estas pruebas son:

- El descubrimiento de dispositivos a través del servicio ssdpsrv proporcionado por Windows.
- Probar la integración de los módulos implementados.
- La gestión de una colección de dispositivos de objetos,
- Mostrar información acerca de los elementos de un dispositivo de objeto,
- Control de dispositivos a través de acciones,
- La recepción de eventos generados por los servicios de notificaciones,
- Mostrar y valorar el tiempo de respuesta de la aplicación comprendido entre su arranque y su descubrimiento en el *Gestor UPnP*.
- Mostrar y valorar el tiempo de repuesta de la aplicación comprendido entre la solicitud de la acción a un dispositivo y la confirmación del éxito de la misma.
- Visualizar el documento de descripción.
- Gestión de los puertos utilizados por ssdpsrv servicio.

Cabe aclarar que las pruebas fueron realizadas con un simulador de bombilla UPnP con opciones de apagado y encendido, también se puede graduar la

intensidad de luz de la bombilla, junto con algunas aplicaciones mediaplayer como el Windows Media Player.

4.1.1. Escenarios de pruebas

En los siguientes apartes se muestran algunas figuras y se describen los escenarios que se utilizaron para realizar las pruebas:

Escenario 1: Se realizo una prueba en una red local implementando tanto el Gestor UPnP, como los dispositivos UPnP sobre un mismo computador portátil , escenario en el cual los tiempos de búsqueda e intercambio de mensajes fueron muy cortos, además de encontrar el total de dispositivo presentes al interior de la red y la administración por parte del Gestor UPnP se realizo exitosamente.

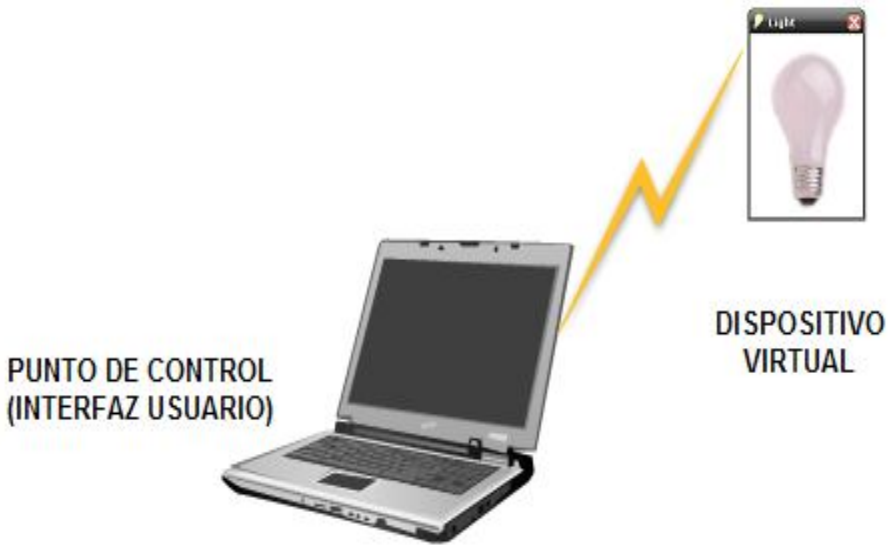


Figura 39. Escenario de Pruebas en un Portátil

FUENTE: Autores

Escenario 2: Para la segunda prueba se creó una red compuesta por un computador portátil el cual hacia la veces de Gestor UPnP conectado a un PC de escritorio, el cual funcionaba como dispositivo UPnP, conectados mediante cable cruzado, para esta prueba los resultados de la búsqueda de los dispositivos se efectuaron en un tiempo razonable, permitiendo controlar los dispositivos; es de aclarar que se podían activar no solo un dispositivo sino varias bombillas y las encontraba todas.

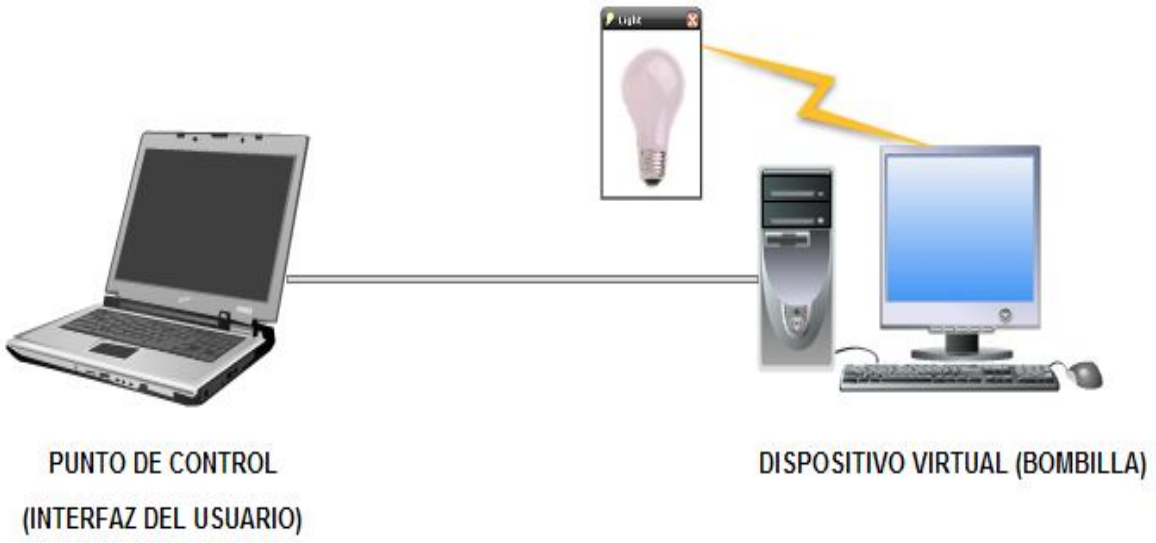


Figura 40. Escenario de Pruebas red Portátil-Desktop

FUENTE: Autores

Escenario 3: Se realizo otra prueba con el computador portátil conectado a toda la sala de proyectos por cable cruzado, donde se demoraba más en encontrar la bombilla pero al final la encontraba y se podía apagar, prender y graduar su intensidad de brillo.

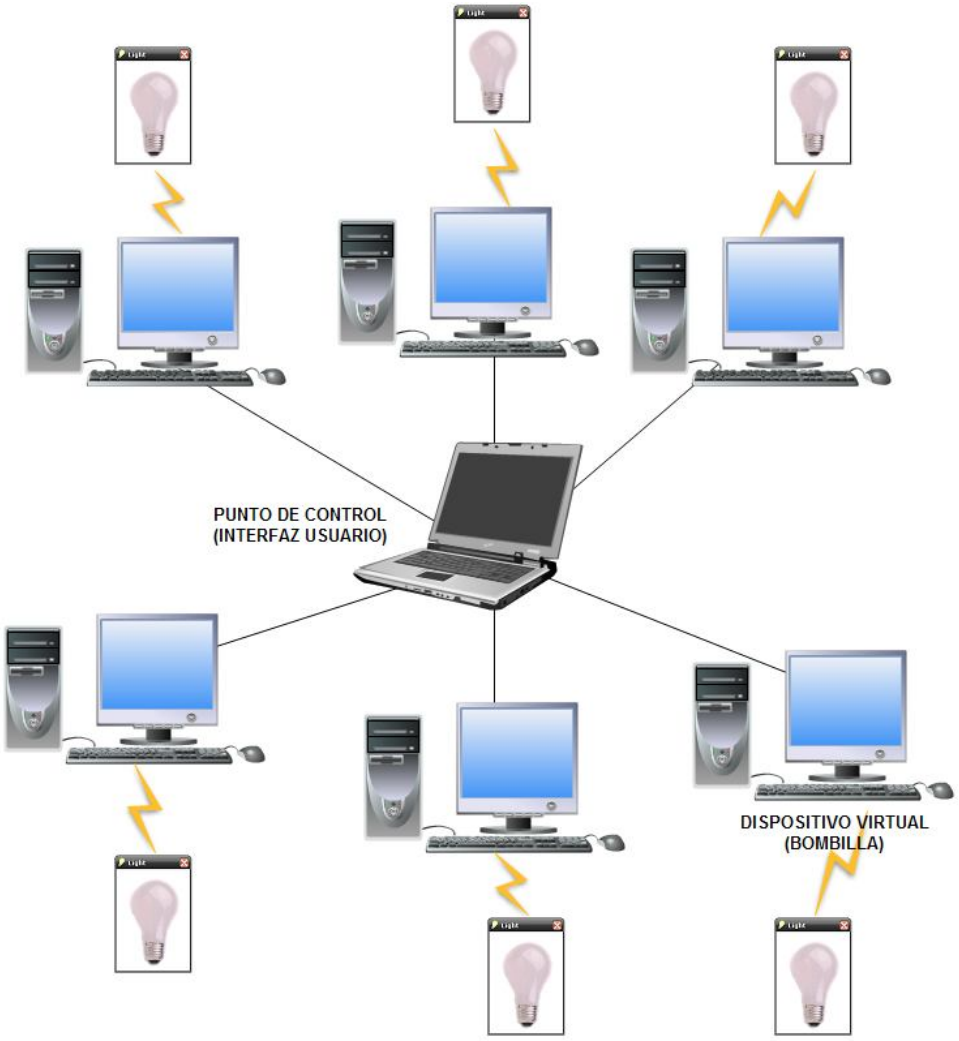


Figura 41. Escenario de Pruebas Red Sala de proyectos

FUENTE: Autores

Escenario 4: También se hizo una prueba con el computador portátil conectado a una red inalámbrica conectada mediante un router inalámbrico, escenario en el cual se encontraba la mayoría de los dispositivos no todos, en un tiempo no tan corto, incluso encontraba al router, y mediante intervención de este enviábamos mensaje de apagado al Windows media player de los portátiles de otros.

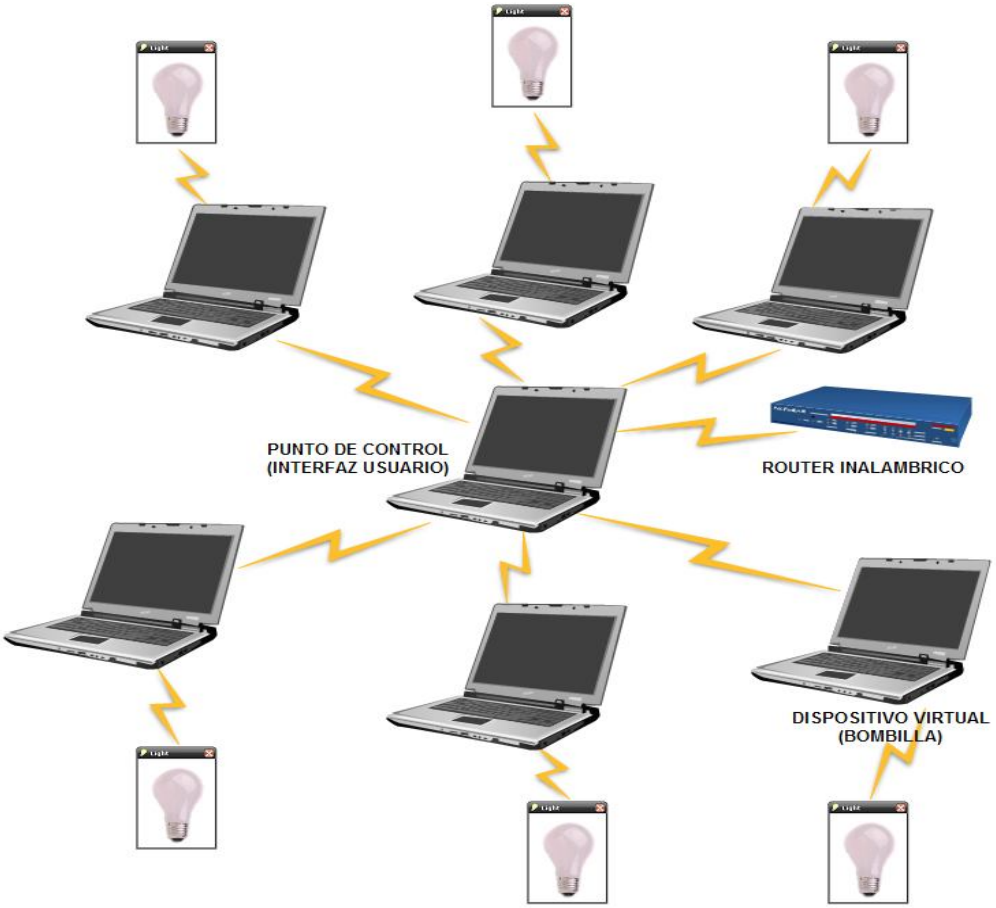


Figura 42. Escenario de Pruebas Red inalámbrica con router

FUENTE: Autores

4.1.2. Características del escenario

El escenario está compuesto por alrededor de 5 equipos los cuales se encuentran conectados en red; las características de los equipos son las siguientes:

1 PC Portátil: Tiene sistema operativo Windows 7. Procesador Intel core 2 duo 2.1 GHz y 4 GB de RAM.

4 Pc`s: Tienen sistema operativo Windows Xp ,3 de ellos tienen procesador 2.8 GHz y 512 MB de RAM, otro tiene procesador de 2.6 GHz y 256 MB de RAM.

4.1.3. Resultados Cualitativos

A continuación se presentan algunos pantallazos obtenidos en la realización de las pruebas realizadas para la tercera versión del gestor de dispositivos UPnP para verificar que presenta la funcionalidad esperada:

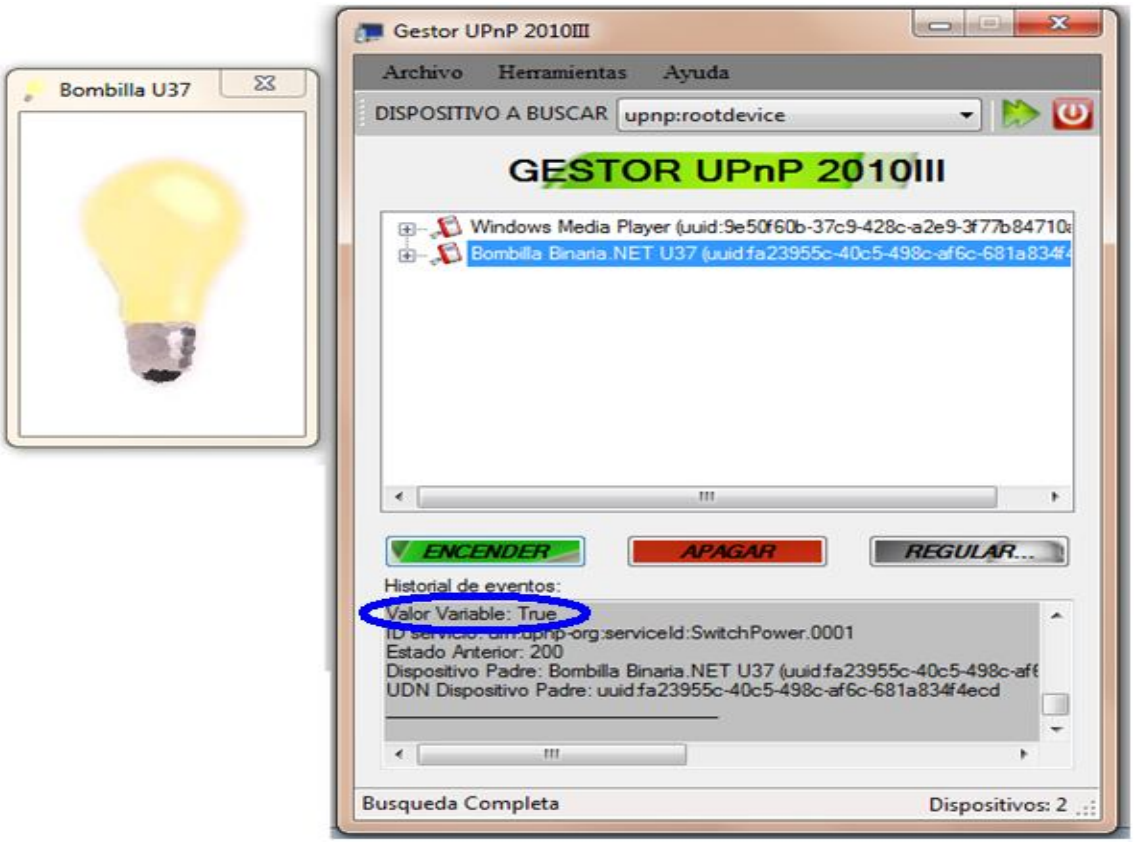


Figura 43. Pantallazo Encendido Dispositivo Bombilla

FUENTE: Autores

En la anterior figura se puede observar la interfaz del usuario, en ella se puede observar que el gestor de dispositivos encontro el windows media player que se encontraba funcionando en ese momento, y tambien encontro una bombilla que se habia activado; el windows media player tambien es considerado como un dispositivo UPnP dentro del forum UPnP.

Bueno, a la bombilla se le ha dado orden de “ENCENDER” y la bombilla prende; dentro de la interfaz podemos observar el historial de eventos donde se puede observar el estado actual de la bombilla, que en este caso es “true” o sea encendido que es además el 100% de luminosidad.

En la siguiente grafica se observa que la bombilla tiene menor cantidad de luminosidad, esto se debe a que se ha elegido el botón “REGULAR”, que sirve para controlar el porcentaje de luz en la bombilla, se puede observar que se encuentra a un 75% de su capacidad.

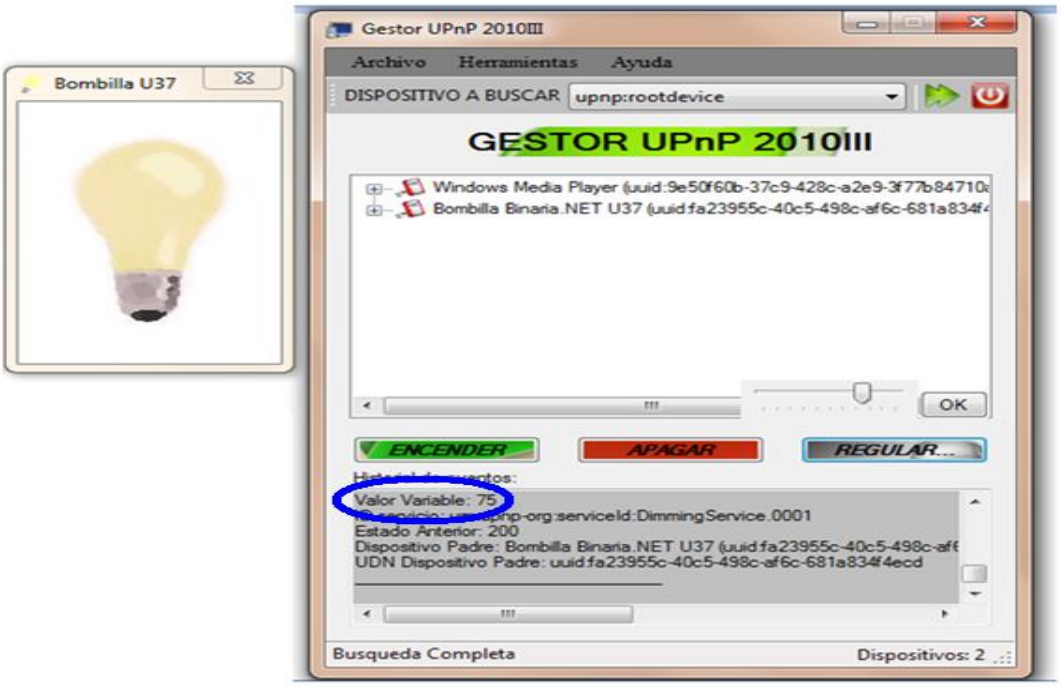


Figura 44. Pantallazo Regulación Luminosidad Dispositivo Bombilla

FUENTE: Autores

Para el apagado de la bombilla, se pude ver que en el historial de eventos, el valor de la variable es “False” ó sea apagado, esta orden fue dada desde el botón “APAGAR”.

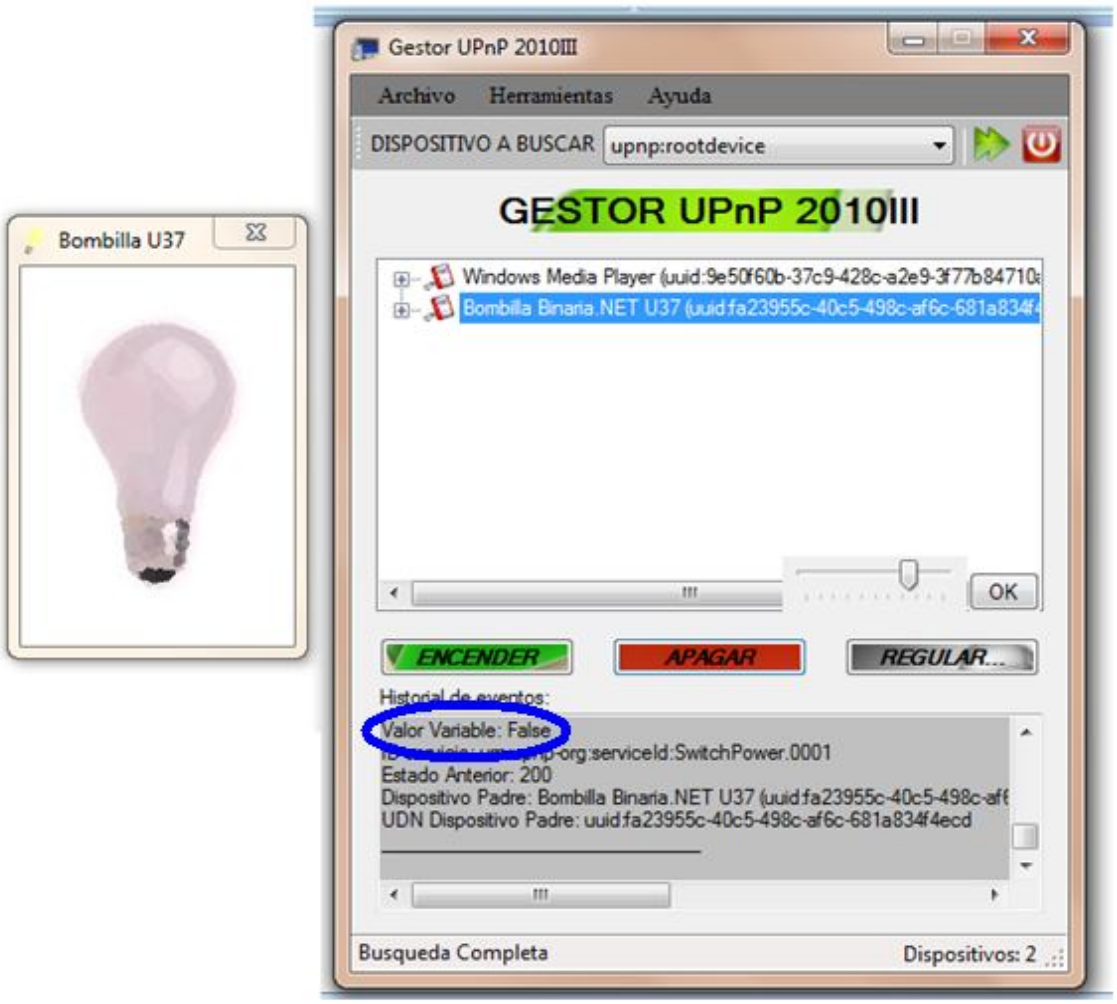


Figura 45. Pantallazo Apagado Dispositivo Bombilla

FUENTE: Autores

Ahora observaremos las pruebas realizadas con el Windows media player; como se puede observar, el gestor de dispositivos encontró este dispositivo y se

encuentra dentro de su listado de dispositivos encontrados, en esta ocasión vamos a “apagar” el media player a través del botón de orden de apagado, se puede observar que aparece la barra de graduar el volumen y se observa en el estado mínimo de capacidad.

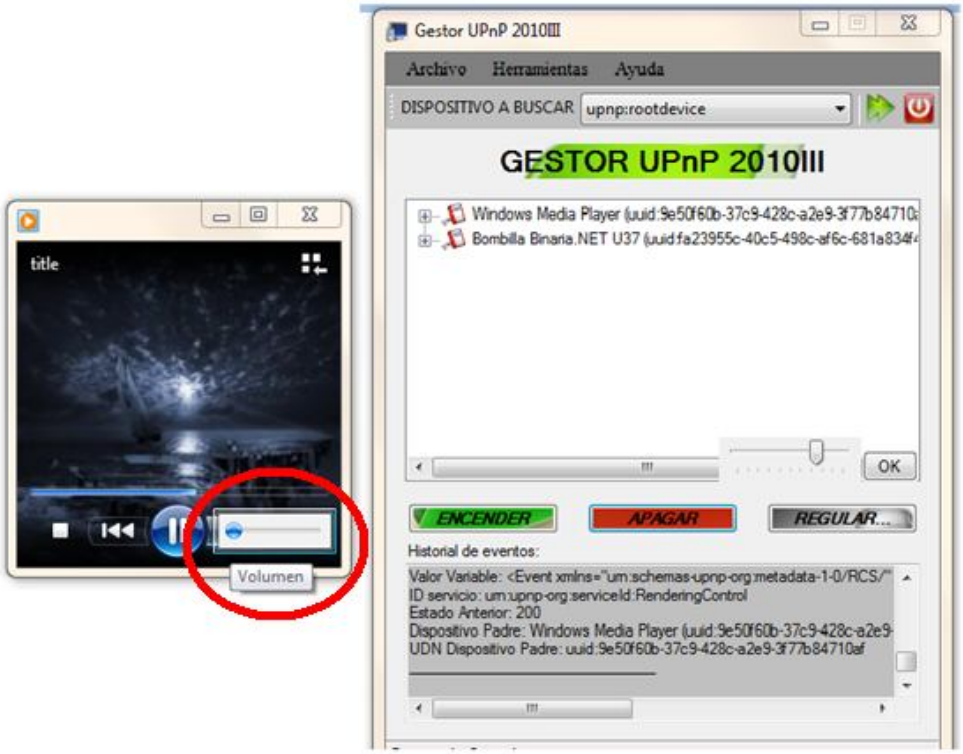


Figura 46. Pantallazo Apagado Volumen Aplicación Media Player

FUENTE: Autores

Sin embargo el volumen en el Windows media player también se puede graduar, en la siguiente imagen vemos la misma barra de control de volumen del media player, el cual está siendo controlado por la barra reguladora del gestor de dispositivos UPnP; y en la otra figura vemos que se ha puesto el máximo volumen al media player.

Figura 47. Pantallazo Regulación Intensidad Volumen Aplicación Media Player

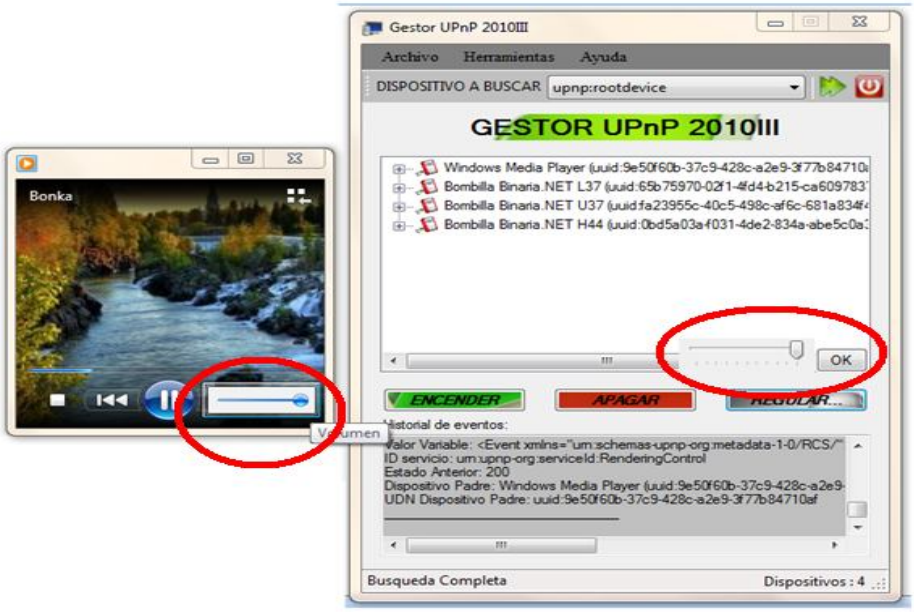


Figura 48. Pantallazo Regulación al 100% intensidad de volumen Media Player

FUENTE FIGURAS 47 Y 48: Autores

En la siguiente figura se muestra la interfaz donde se se muestra la descripcion del dispositivo, como por ejemplo el nombre del dispositivo, la URL que maneja, el modelo al cual corresponde, el nombre del fabricante, numero de serial.

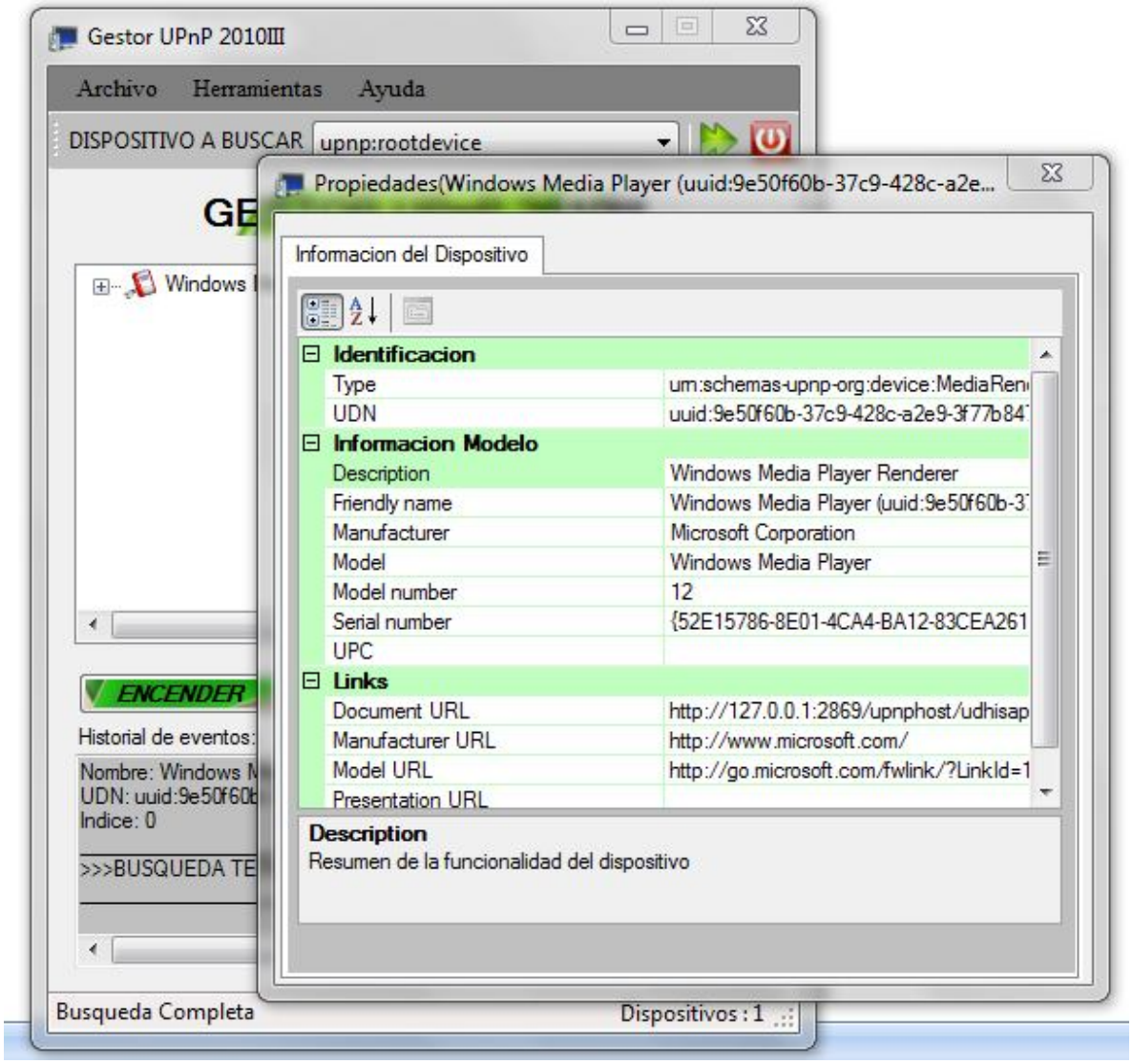


Figura 49. Pantallazo Información Técnica Aplicación Media Player

FUENTE: Autores

En la siguiente figura se muetra un listado de los servicios que puede prestar el dispositivo, en este caso el windows media player , como son el getvolume que sirve para obtener innformacion del estado actual del volumen, el setvolume que sirve para prender o apagar el media player, o dar un valor de 0 a 100 donde 0 es apagado y 100 es el valor de maxima volumen. Estos servicios son configurados por el fabricante , es decir , el gestor de dispositivos se informa de los servicios ofrecidos por el dispositivo.

Tambien se puede observar que en la barra principal, podemos desplegar la opcion herramientas, que es desde donde podemos abrir y cerrar los puerto usados por el procoloco UPnP.

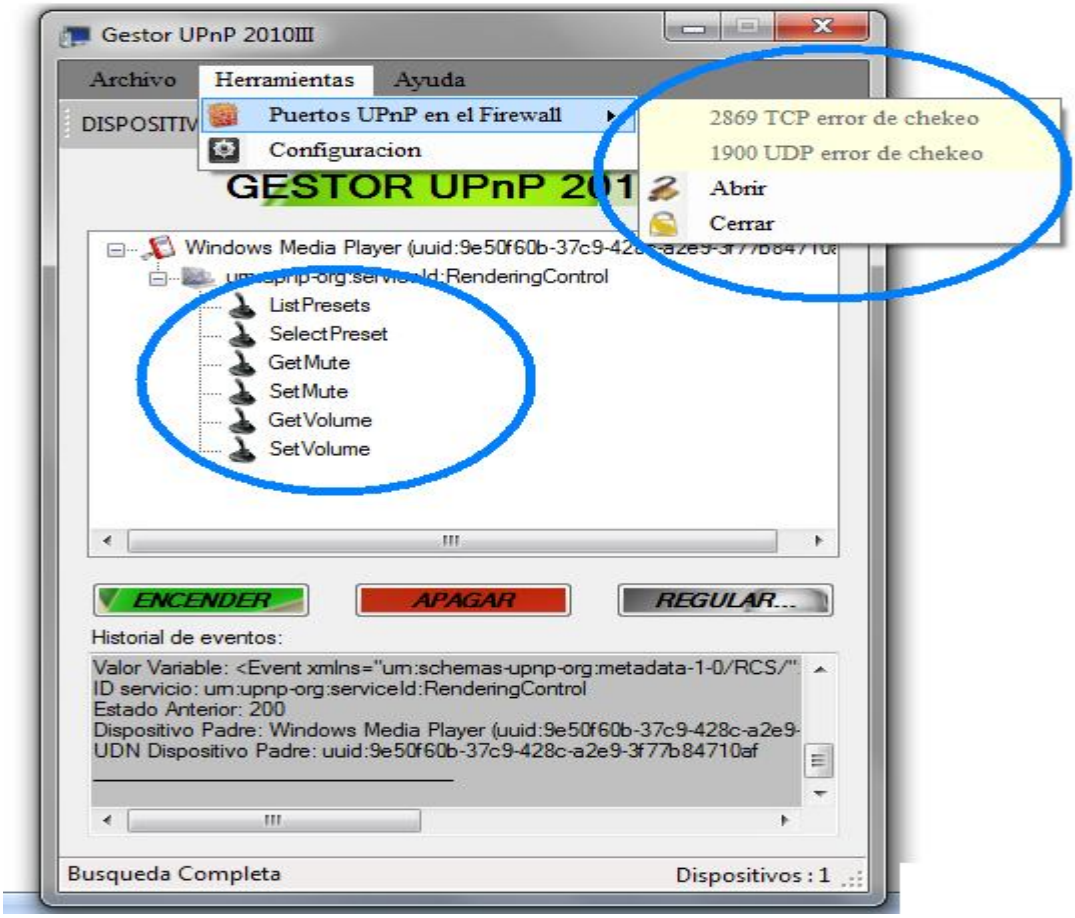


Figura 50. Pantallazo Funcionalidades Gestor UPnP

FUENTE: Autores

Ahora en la siguiente figura, se podra observar la interfaz del gestor de dispositivos, pero en esta ocacion no solo encontro el windows media player sino que ademas tambien ha encontrado dos bombillas con las cuales en una de ellas se ha encendido normalmente y en la otra señalada donde su intensidad es diferente a la otra bombilla.

Figura 51. Pantallazo Funcionamiento Gestor UPnP

FUENTE: Autores

A continuación se presentan un par de pantallazo de la actualización del Gestor de Dispositivos UPnP, en la cual se incluyo la funcionalidad de activar o desactivar el Modo Colaboracion. Funcionalidad que permite que un dispositivo tipo Switch interactué con un Dispositivo tipo Bombilla, mediante la intervención del Gestor, ya que es este el que especifica la bombilla sobre la cual recaen las acciones del Switch, cumpliendo asi la parte de colaboración entre dispositivos planteada como parte del objetivo general del actual proyecto.

Figura 52. Activación del Modo Colaboración

FUENTE: Autores

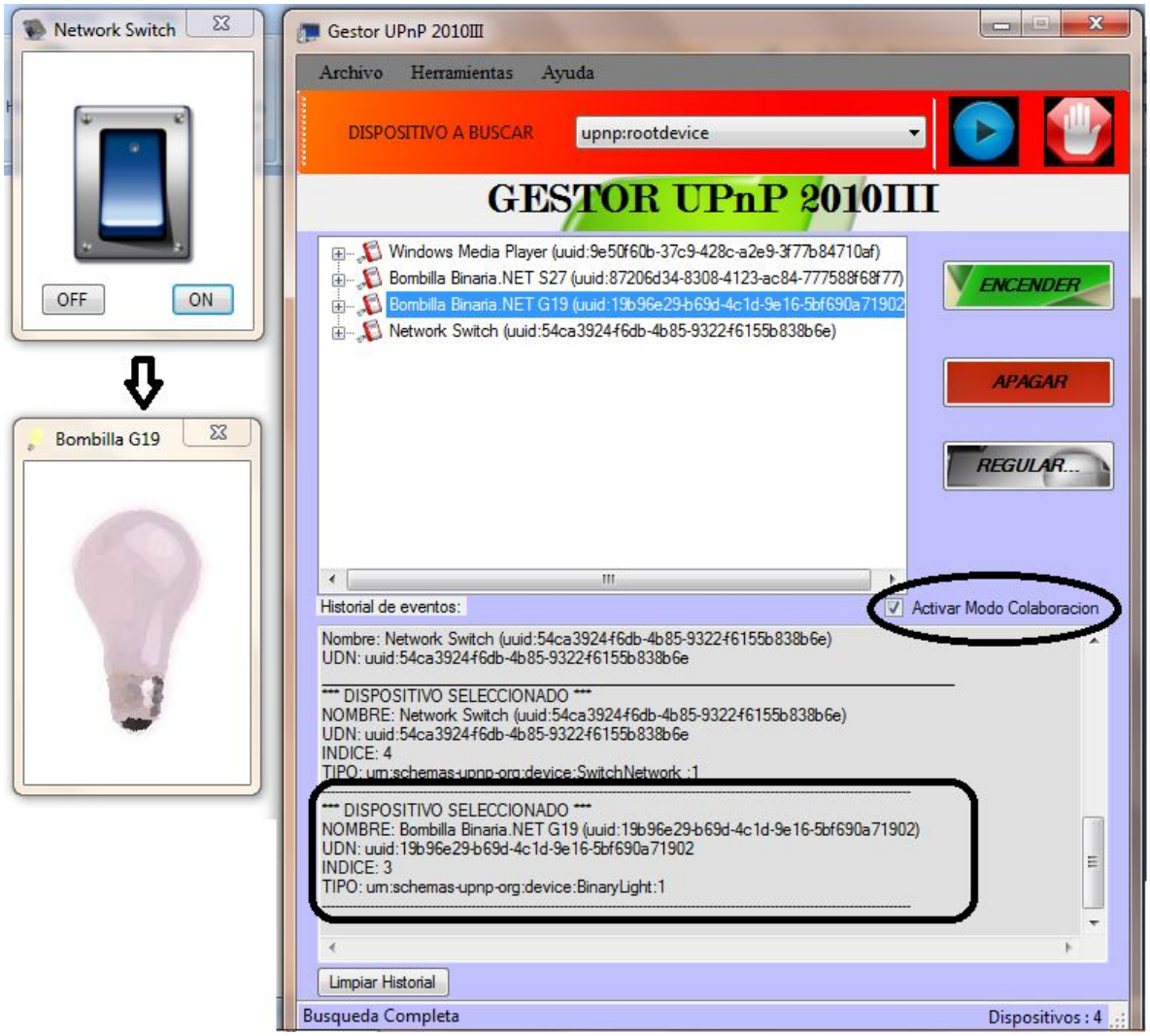


Figura 53. Proceso de Interacción Switch-Bombilla

FUENTE: Autores

4.1.4. Presentación de Resultados Cuantitativos

A continuación se muestra una tabla de valores de los tiempos promedio que el gestor de dispositivos se demora encontrando los dispositivos en cada uno de los escenarios anteriormente planteados.

Para la toma de los tiempos empleados para el descubrimiento e intercambio de mensajes se emplearos tres dispositivos, los cuales fueron dos simuladores tipo Bombilla UPnP y una aplicación UPnP como lo es el Windows Media Player 11.

Tiempos medios empleados para la búsqueda:

Escenario 1 (Prueba dentro del mismo PC portátil)

Tiempo gastado en encontrar el primer dispositivo	0.82 seg
Tiempo gastado en encontrar el segundo dispositivo	1.16 seg
Tiempo gastado en encontrar el tercer dispositivo	1.51 seg

* Encontrados el 100% de dispositivos disponibles en el equipo.

* El tiempo empleado en el intercambio de mensajes en irrelevante ya que estos se reciben de manera instantánea.

Escenario 2 (Prueba con tres equipos diferentes en una red cableada)

Primer ensayo (tomado del equipo más rápido):

Tiempo gastado en primer dispositivo	40 seg
Tiempo gastado en segundo dispositivo	No encontrado
Tiempo gastado en tercer dispositivo	No encontrado

Encontró la bombilla de otro equipo pero las demás no las encontró.

Segundo ensayo (tomado del equipo más rápido):

Tiempo gastado en primer dispositivo	44 seg
Tiempo gastado en segundo dispositivo	No encontrado
Tiempo gastado en tercer dispositivo	No encontrado

Tercer ensayo (tomado del equipo más rápido):

Tiempo gastado en primer dispositivo	1,5 seg

Tiempo gastado en segundo dispositivo	15 seg
Tiempo gastado en tercer dispositivo	20 seg
Tiempo gastado en cuarto dispositivo	25 seg

Encontrados el 100% de dispositivos disponibles en la red en este tercer ensayo

Se utilizo en uno de los equipos un procesador intel core 2 duo de 2.2 GHz, mas rápido que los demás, y bueno, realmente se observo que encontraba mayor numero de dispositivos y más rápido que en los otros equipos de red, las acciones que se emitían se realizaban prácticamente al instante, en un segundo; aunque no siempre pasaba lo mismo, esto debido a la congestión en la red, en ocasiones encontraba primero los dispositivos instalados en otros equipos en comparación con los del mismo computador, en ocasiones no encontraba a totalidad de los dispositivos que se encontraban en la red.

Prueba con dos equipos diferentes en una red inalámbrica.

Primer ensayo (tomado del equipo más rápido):

Tiempo gastado en primer dispositivo	2.13 seg
Tiempo gastado en segundo dispositivo	43.07 seg

Tiempo gastado en tercer dispositivo	1.26 min

Segundo ensayo (tomado del equipo más rápido):

Tiempo gastado en primer dispositivo	1.8 seg
Tiempo gastado en segundo dispositivo	42.8 seg
Tiempo gastado en tercer dispositivo	1.24 min

Se pone lenta la red, se demora en encontrarlos.

Tercer ensayo (tomado del equipo más rápido):

Tiempo gastado en primer dispositivo	1.32 min
Tiempo gastado en segundo dispositivo	1.99 min
Tiempo gastado en tercer dispositivo	1.32 min

Se observa que en la red inalámbrica se hace más lento los resultados de búsqueda.

Sin embargo hubo un problema, no encontraba los dispositivos del otro equipo y viceversa, tal vez por alguna configuración en el router inalámbrico, los resultados son de los dispositivos instalados en el mismo computador.

Para la toma de los datos de los ensayos tuvimos las dos siguientes condiciones:

- Se considerarán nulos los retardos de transmisión, ya que el propósito de estas pruebas no incluye la valoración ni del tipo ni del estado de la red.

- Los resultados obtenidos de dichas pruebas tendrán un carácter orientativo, ya que en todos los procesos intervienen múltiples factores asociados a las capacidades de computación de los equipos que intervienen.

5. CONCLUSIONES

En este capítulo se recogen las conclusiones obtenidas de la realización de este proyecto, en primer lugar se detalla el grado de cumplimiento de los objetivos planteados.

- Como objetivo general de este proyecto se estableció que se desarrollaría un prototipo de gestor de dispositivos UPnP que se integraran a una red, dando como resultado una aplicación C# y C++ para la plataforma Windows, que permite el descubrimiento de dispositivos que manejen el protocolo UPnP y a su vez permite cierto control sobre estos dispositivos atreves de acciones estipuladas por la información del fabricante del dispositivo conforme al estándar UPnP.

- Para llevar a cabo este proyecto se definieron primero los diferentes módulos que se desarrollarían en base a los objetivos propuestos, fue necesario investigar a profundidad para documentarse lo suficiente y de esta forma saber o conocer el funcionamiento del protocolo UPnP, además de analizar las interfaces que proporciona Microsoft para el desarrollo de este gestor, pues fueron ellos los que fundaron este protocolo y también son quienes informan de cuáles son las funciones normalmente usadas de acuerdo a cada interfaz y al uso que se requiera.

- Se comprobó el correcto funcionamiento del prototipo en un escenario real y se definieron las limitaciones derivadas de su diseño, como la imposibilidad de interactuar con mas de un dispositivo a la vez. Ademas se realizaron varias pruebas en escenarios reales comprobando principalmente la correcta comunicación del dispositivo con el Gestor de Dispositivos y el correcto procesamiento de la invocación de las acciones. Se tomaron varias

medidas de tiempos de respuesta de la aplicación para su valoración en esta temática y así poder mostrar la eficiencia de la misma.

- Se observo que cuando se ponía a funcionar el gestor de dispositivos a través de una red de varios equipos, la aplicación se demoraba en encontrar los dispositivos, esto se debe en parte al tipo y al estado de la red, en otras palabras, la funcionalidad de la aplicación puede verse afectada por el estado de la red.

- Se pudo observar el servicio que puede prestar esta aplicación, principalmente al entrenamiento digital en el hogar; las tecnologías aplicadas a este tipo de ocio evolucionan a un ritmo vertiginoso y era la oportunidad de introducirnos en el desarrollo de aplicaciones basadas en ellas; en este caso ha sido UPnP. Es fascinante la facilidad con la que un usuario puede disponer de una red de contenidos multimedia compartidos en la que se puede controlar su reproducción en dispositivos remotos desde un único dispositivo controlador.

- Es lamentable que no se haya podido realizar pruebas con dispositivos físicos para las pruebas inalámbricas, pues es relativamente nueva la implementación de este protocolo en la tecnología hardware existente en nuestro mercado, sin embargo ya desde hace algún tiempo viene incorporada en algunos servidores y routers según información del fórum UPnP.

6. RECOMENDACIONES

- ✓ Para el correcto funcionamiento del software es necesario habilitar los puertos de comunicación UPnP en el firewall del PC, ya sea de manera manual o mediante la misma aplicación para que no ocurra impedimentos al realizar la búsqueda, también desactivar temporalmente el corta fuegos incluido en algunos antivirus, pues impiden el proceso necesario para el reconocimiento de los dispositivos.

- ✓ También es importante activar en el firewall los permisos necesarios para que los dispositivos UPnP acepten y envíen información a la red ya que de lo contrario los mensajes de anunciamiento serán bloqueados y por lo tanto los dispositivos no serán encontrados por el gestor.

- ✓ Se recomienda para futuras versiones del software desarrollarlo en plataformas que permitan su implementación en dispositivos móviles como teléfonos celulares, Palms o Pocket PC para mayor comodidad en el manejo del mismo.

- ✓ El software actual está orientado básicamente a controlar tres tipos de acciones, encendido, apagado, y regulación de una bombilla binaria y el volumen en el reproductor de audio, por lo que para futuras versiones sería recomendable desarrollar la programación de las funcionalidades correspondientes para los demás dispositivos UPnP disponibles según el estándar dado por el Foro UPnP.

7. BIBLIOGRAFIA

[1] STEVENS, Perdita; POOLEY, Rob**.** Utilización de UML en ingeniería del software con objetos y componentes. Madrid: Pearson Educación, 2002. 291 p.

[2] FOXALL, James. El libro de visual C# 2005. Madrid: Anaya Multimedia, 2007. 558 p.

[3] GALLO, Michael A.; HANCOCK, William M. Comunicación entre computadoras y tecnologías de redes. Mexico: Thomson. 2002. 632 p.

[4] KINGSLEY HUGHES, Adrian; KINGSLEY HUGHES, Kathie. C# 2005. Madrid: Anaya Multimedia. 2007. 448 p.

[5] FOWLER, Martin; SCOTT, Kendall. UML GOTA A GOTA. México: Pearson Educación: Addison Wesley Longman. 1999. 203 p.

[6] Sección del sitio oficial del Foro UPnP sobre la especificación para seguridad. http://www.upnp.org/standardizeddcps/security.asp

[7] Documento técnico sobre la tecnología UPnP realizado por Microsoft. http://www.upnp.org/download/UPNP_UnderstandingUPNP.doc

[8] Sección del sitio oficial del Foro UPnP sobre la especificación para la calidad de servicio (QoS). http://www.upnp.org/specs/qos/

[9] Sitio web oficial del Foro UPnP. http://www.upnp.org/

[10] Definición oficial de la plataforma de programación Visual Studio C#.

http://msdn.microsoft.com/es-es/library/z1zx9t92(VS.80).aspx

[11] Características de la plataforma de programación Visual Studio C#.
http://www.clikear.com/manuales/csharp/c10.aspx

[12] Ventajas de la plataforma de programación Visual Studio C#.
http://urriellu.net/es/articles-software/csharp-advantages.html

[13] Definición del componente software Microsoft .NET Framework 3.5.
http://www.alegsa.com.ar/Dic/.net%20Framework.php

[14] Características del componente software Microsoft .NET Framework 3.5.
http://www.taringa.net/posts/downloads/1842973/Microsoft-_NET-Framework-3_5.html

[15] Ventajas del componente software Microsoft .NET Framework 3.5.
http://es.answers.yahoo.com/question/index?qid=20070817175348AAgSFgE

[16] Documento informativo sobre UPnP de la revista digital CASADOMO
http://www.casadomo.com/noticiasDetalle.aspx?c=25&m=29&idm=32&pat=148&n2=148

[17] Página web de “libupnp” (UPnP SDK).
http://pupnp.sourceforge.net/

[18] Descripción de las características de UPnP en Windows XP.
http://support.microsoft.com/kb/323713

[19] Ajustes en la configuración del protocolo UPnP.
http://msdn2.microsoft.com/en-us/library/aa381091(VS.85).aspx

[20] Como afecta el firewall de Windows al protocolo UPnP.
http://support.microsoft.com/kb/886257"

[21] Definicion Computacion Ubicua.
http://es.wikipedia.org/wiki/Computaci%C3%B3n_ubicua

INDICE

ANEXOS

ANEXO I: ESPECIFICACIÓN DE LAS INTERFACES TENIDAS EN CUENTA PARA EL DESARROLLO DEL GESTOR UPNP

Las interfaces siguientes son parte de la API de puntos de control con la tecnología UPnP. El API de Punto de Control es compatible con Microsoft Visual Basic Scripting Edition (VBScript), Microsoft Visual Basic y C + +.

A. IUPnPAddressFamilyControl Interfaz

La interfaz de acceso a la bandera IUPnPAddressFamilyControl familia de direcciones del dispositivo objeto del Finder.

Cuando implementar:

No se debe implementar esta interfaz. La implementación de Microsoft estándar proporciona una funcionalidad completa.

Cuándo utilizar:

Utilice esta interfaz para controlar el filtrado de dispositivos encontrados. La aplicación establece el indicador de familia de direcciones de un objeto buscador de dispositivos antes de comenzar una búsqueda. La solicitud será notificada sólo

acerca de los dispositivos que tienen direcciones IP que son de la familia de direcciones especificada.

El alcance de este ajuste es por dispositivo objeto del Finder. Por lo tanto, una aplicación puede tener múltiples dispositivos objetos Buscador de instancias, cada una con un valor diferente en su pabellón del domicilio familiar. Cada objeto de dispositivos Finder devolverá el subconjunto adecuado de estos dispositivos si la aplicación utiliza el objeto de hacer una búsqueda.

Usted puede hacer una consulta para esta interfaz desde la interfaz de IUPnPDeviceFinder.

Miembros:

La interfaz de IUPnPAddressFamilyControl hereda de la interfaz IUnknown. IUPnPAddressFamilyControl también define los siguientes tipos de miembros:

Métodos:
La interfaz IUPnPAddressFamilyControl define los siguientes métodos.

Método Descripción:
GetAddressFamily Recupera el valor actual de la bandera de la familia dirección del objeto de dispositivo Finder.

SetAddressFamily Activa el indicador de familia de direcciones del dispositivo objeto del Finder.

Requerimientos:

Mínimos admitidos cliente
Windows Vista
Mínimos admitidos servidor
Ninguno apoyo
Encabezamiento
UPnP.h
DLL
UPnP.dll

B. IUPnPDescriptionDocument Interfaz

La interfaz permite que una aplicación IUPnPDescriptionDocument para cargar una descripción del dispositivo.

Cuando implementar:

No se debe implementar esta interfaz. La implementación de Microsoft estándar proporciona una funcionalidad completa.

Cuándo utilizar:

Debe utilizar esta interfaz al crear una página Web que se ejecuta desde un dispositivo.

Nota: para C + +, se recuperan todas las propiedades por un método que se llama anteponiendo get_ al nombre de la propiedad. Por ejemplo, si el nombre de la propiedad es SomeState, entonces get_SomeState recupera la propiedad.

Miembros:

La interfaz de IUPnPDescriptionDocument hereda de la interfaz IDispatch. IUPnPDescriptionDocument también define los siguientes tipos de miembros:

Métodos
Propiedades
Métodos
La interfaz IUPnPDescriptionDocument define los siguientes métodos.

Método Descripción
Load carga de un documento de forma sincrónica.

LoadAsync Carga un documento de forma asincrónica.

Abort Detiene una operación de carga asincrónica.

Devuelve RootDevice el dispositivo raíz del árbol de dispositivos del documento cargado.

DeviceByUDN Devuelve un dispositivo a partir del documento actualmente cargado con el nombre de dispositivo especificado único (UDN).

Propiedades:

La interfaz IUPnPDescriptionDocument define las siguientes propiedades.

Propiedad acceso de tipo descripcion

LoadResult

Resultados de sólo lectura de código que indica el éxito o el fracaso de una operación de carga completa.

ReadyState

Condición Jurídica y Social de sólo lectura de la operación de carga de documentos.

Requerimientos:

Mínimos admitidos cliente

Windows XP

Mínimos admitidos servidor

Ninguno apoyo

Encabezamiento

UPnP.h

UPnP.dll

C. IUPnPDescriptionDocumentCallback Interfaz

La interfaz permite IUPnPDescriptionDocumentCallback el marco UPnP para comunicar los resultados de una operación de carga asincrónica a una aplicación.

Cuando implementar:

Los desarrolladores deben implementar esta interfaz para aplicaciones C + + que se escriben usando los métodos de carga asíncrona de la interfaz IUPnPDescriptionDocument. Los desarrolladores que utilizan todos los lenguajes

de programación, como Visual Basic, debe implementar una función de devolución de llamada personalizados para recibir devoluciones de llamada.

Cuándo utilizar:

El entorno UPnP usa esta interfaz para notificar a la aplicación cuando la operación de carga se ha completado.

Miembros:

La interfaz de IUPnPDescriptionDocumentCallback hereda de la interfaz IUnknown. IUPnPDescriptionDocumentCallback también define los siguientes tipos de miembros:

Métodos
Métodos
La interfaz IUPnPDescriptionDocumentCallback define el método siguiente.

Método Descripción
LoadComplete Se invoca por el entorno UPnP para notificar la aplicación de que una operación de carga se ha completado.

Requerimientos:

Mínimos admitidos cliente
Windows XP
Mínimos admitidos servidor
Ninguno apoyo
Encabezamiento

UPnP.h

DLL

UPnP.dll

D. IUPnPDevice Interfaz

La interfaz permite que una aplicación IUPnPDevice para recuperar información sobre un dispositivo específico.

Cuando implementar:

No se debe implementar esta interfaz. La implementación de Microsoft estándar proporciona una funcionalidad completa.

Cuándo utilizar:

Utilice esta interfaz cuando se necesita acceder a las propiedades y servicios básicos de un dispositivo.

Nota: para C + +, se recuperan todas las propiedades por un método que se llama anteponiendo get_ al nombre de la propiedad. Por ejemplo, si el nombre de la propiedad es SomeState, recupera get_SomeState la propiedad.

Miembros:

La interfaz de IUPnPDevice hereda de la interfaz IDispatch. IUPnPDevice también define los siguientes tipos de miembros:

Métodos

Propiedades

Métodos

La interfaz IUPnPDevice define el método siguiente.

Método Descripción

IconURL Devuelve una URL desde la que puede ser un icono del formato especificado cargado.

Propiedades

La interfaz IUPnPDevice define las siguientes propiedades.

Propiedad: Acceso de tipo, descripción.

Children: Dispositivos secundarios del dispositivo.

Description: Forma humana en formato electrónico del resumen de la funcionalidad de un dispositivo.

FriendlyName: Nombre del dispositivo para mostrar.

HasChildren: Indica si el dispositivo tiene cualquier niño dispositivos

IsRootDevice: Indica si el dispositivo es el dispositivo superior en el árbol de dispositivos.

ManufacturerName: Forma legible el nombre del fabricante.

ManufacturerURL : Dirección del sitio Web del fabricante.

ModelName: Forma legible el nombre del modelo.

ModelNumber : Forma legible del número de modelo.

ModelURL: URL de una página Web que contiene información específica del modelo.

ParentDevice: Padre del dispositivo.

PresentationURL: Presentación URL de una página Web que se puede utilizar para controlar el dispositivo

RootDevice: Dispositivo superior en el árbol de dispositivos..

SerialNumber: Forma legible por humanos del número de serie..

Services : Lista de los servicios prestados por el dispositivo.

Type: Identificador de recursos uniforme (URI) para el tipo de dispositivo..

UniqueDeviceName: Nombre único dispositivo (UDN) del dispositivo.

UPC : Forma legible el código de producto.

Requerimientos:

Mínimos admitidos cliente
 Windows XP
Mínimos admitidos servidor
 Ninguno apoyo
Encabezamiento
 UPnP.h

DLL

UPnP.dll

E. IUPnPDeviceDocumentAccess Interfaz

La interfaz permite IUPnPDeviceDocumentAccess una solicitud para obtener la dirección del documento de descripción del dispositivo.

Nota: esta interfaz no admite secuencias de comandos.

Cuando implementar:

No se debe implementar esta interfaz. La implementación de Microsoft estándar proporciona una funcionalidad completa.

Cuándo utilizar:

Utilice esta interfaz para obtener específico del dispositivo y las propiedades específicas de la función-que no se exponen por la interfaz IUPnPDevice.

Usted puede hacer una consulta para esta interfaz desde la interfaz de IUPnPDevice.

Miembros:

La interfaz de IUPnPDeviceDocumentAccess hereda de la interfaz IUnknown. IUPnPDeviceDocumentAccess también define los siguientes tipos de miembros:

Métodos

Métodos

La interfaz IUPnPDeviceDocumentAccess define el método siguiente.

Método Descripción

GetDocumentURL Devuelve la URL desde la cual puede ser el documento de descripción de dispositivo cargado.

Requerimientos:

Mínimos admitidos cliente

Windows XP

Mínimos admitidos servidor

Ninguno apoyo

Encabezamiento

UPnP.h

DLL

UPnP.dll

F. IUPnPDeviceDocumentAccessEx Interfaz

La interfaz proporciona IUPnPDeviceDocumentAccessEx un método para obtener todo el dispositivo de documentos XML para la descripción de un dispositivo específico.

Cuando implementar:

Microsoft proporciona una implementación completa de esta interfaz.

Miembros:

La interfaz de IUPnPDeviceDocumentAccessEx hereda de la interfaz IUnknown. IUPnPDeviceDocumentAccessEx también define los siguientes tipos de miembros:

Métodos

La interfaz IUPnPDeviceDocumentAccessEx define el método siguiente.

Método GetDocument Recupera el dispositivo de documentos XML descripción de un dispositivo UPnP.

Comentarios:
Esta interfaz se obtiene llamando a QueryInterface en el mismo objeto que proporciona una implementación de IUPnPDevice, tras lo cual se puede llamar getDocument en él.

Requerimientos:

Mínimos admitidos cliente
Windows 7
Mínimos admitidos servidor
Ninguno apoyo
Encabezamiento
UPnP.h
DLL
UPnP.dll

G. IUPnPDeviceFinder Interfaz

La interfaz permite que una aplicación IUPnPDeviceFinder para encontrar un dispositivo.

Cuando implementar:

No se debe implementar esta interfaz. La implementación de Microsoft estándar proporciona una funcionalidad completa.

Cuándo utilizar:

Utilice esta interfaz cuando una aplicación necesita la ubicación de un dispositivo específico que necesita para controlar.

Usted puede hacer una consulta para esta interfaz desde la interfaz de IUPnPAddressFamilyControl.

Esta interfaz se puede llamar desde dentro de una página Web.

Miembros:

La interfaz de IUPnPDeviceFinder hereda de la interfaz IDispatch. IUPnPDeviceFinder también define los siguientes tipos de miembros:

Métodos
Métodos

La interfaz IUPnPDeviceFinder define los siguientes métodos.

Método Descripción
Búsquedas FindByType sincrónica para los dispositivos por tipo de dispositivo o tipo de servicio.

CreateAsyncFind Crea una operación de búsqueda asincrónica.

StartAsyncFind Inicia una búsqueda asincrónica.

CancelAsyncFind Cancela una búsqueda asincrónica.

Búsquedas FindByUDN sincrónica de un dispositivo por su nombre de dispositivo único (UDN).

Requerimientos:

Mínimos admitidos cliente
Windows XP
Mínimos admitidos servidor
Ninguno apoyo
Encabezamiento
UPnP.h
DLL
UPnP.dll

F. IUPnPDeviceFinderAddCallbackWithInterface Interfaz

La interfaz permite IUPnPDeviceFinderAddCallbackWithInterface el marco UPnP para comunicarse con una aplicación:

Los dispositivos encontrados durante la búsqueda asincrónica.
La interfaz de red a través del cual llegó el anuncio del dispositivo.

Cuando implementar:

Usted debe implementar esta interfaz, junto con la interfaz IUPnPDeviceFinderCallback para aplicaciones C + + que utilizan los métodos de búsqueda asíncrona de la interfaz IUPnPDeviceFinder y que necesita conocer la interfaz de red a través del cual llegó el anuncio del dispositivo. Si la interfaz de red no es una preocupación, aplicar sólo la interfaz IUPnPDeviceFinderCallback.

Para obtener más información y código de ejemplo, véase la búsqueda asincrónica.

Cuándo utilizar:

Si ha implementado esta interfaz, el entorno UPnP usa para:

Notificar a la aplicación cuando un dispositivo se añade a la red.
Especificar para la aplicación de la interfaz de red a través del cual llegó el anuncio del dispositivo.
De lo contrario, el entorno UPnP usa la interfaz IUPnPDeviceFinderCallback para notificar la aplicación cuando un dispositivo se añade.
El entorno UPnP determina si existe IUPnPDeviceFinderAddCallbackWithInterface al consultar el objeto de devolución de llamada a la que un puntero se pasa en el parámetro de la pUnkCallback IUPnPDeviceFinder:: Método CreateAsyncFind.

Miembros:

La interfaz de IUPnPDeviceFinderAddCallbackWithInterface hereda de la interfaz IUnknown. IUPnPDeviceFinderAddCallbackWithInterface también define los siguientes tipos de miembros:

Métodos
Métodos
La interfaz IUPnPDeviceFinderAddCallbackWithInterface define el método siguiente.

Método Descripción
DeviceAddedWithInterface Se invoca por el marco UPnP para notificar a la aplicación que un dispositivo ha sido añadido a la red.

Comentarios:

Si decide implementar esta interfaz, también debe implementar la interfaz IUPnPDeviceFinderCallback.

Requerimientos:

Mínimos admitidos cliente
Windows Vista
Mínimos admitidos servidor
Ninguno apoyo
Encabezamiento
UPnP.h
DLL

UPnP.dll

G. IUPnPDeviceFinderCallback Interfaz

La interfaz permite IUPnPDeviceFinderCallback el marco UPnP para comunicar los resultados de una búsqueda asincrónica a una aplicación.

Cuando implementar:

Usted debe implementar esta interfaz para aplicaciones C + + que utilizan los métodos de búsqueda asíncrona de la interfaz IUPnPDeviceFinder.

Para obtener más información y código de ejemplo, véase la búsqueda asincrónica.

Cuándo utilizar:

El entorno UPnP usa esta interfaz para notificar a la aplicación cuando un dispositivo se añade o se quita de la red, y cuando la búsqueda inicial se ha completado.

Notificación de un dispositivo añadido: El entorno UPnP puede utilizar la interfaz en lugar de IUPnPDeviceFinderAddCallbackWithInterface IUPnPDeviceFinderCallback para notificar la aplicación cuando un dispositivo se añade a la red. El entorno UPnP se consulta para ver si existe IUPnPDeviceFinderAddCallbackWithInterface. Si es así, el entorno UPnP lo usará para notificar a la aplicación del dispositivo adicional. De lo contrario, el marco UPnP usará la interfaz IUPnPDeviceFinderCallback.

Miembros:

La interfaz de IUPnPDeviceFinderCallback hereda de la interfaz IUnknown. IUPnPDeviceFinderCallback también define los siguientes tipos de miembros:

Métodos:

Métodos
La interfaz IUPnPDeviceFinderCallback define los siguientes métodos.

Método Descripción
DeviceAdded Se invoca por el marco UPnP para notificar a la aplicación que un dispositivo ha sido añadido a la red.

DeviceRemoved Se invoca por el marco UPnP para notificar a la aplicación que un dispositivo se ha quitado de la red.

SearchComplete Se invoca por el marco UPnP para notificar a la aplicación que la búsqueda inicial de los dispositivos de red se ha completado.
Requerimientos:

Mínimos admitidos cliente
Windows XP
Mínimos admitidos servidor
Ninguno apoyo
Encabezamiento
UPnP.h
DLL

UPnP.dll

H. IUPnPDevices Interfaz

La interfaz IUPnPDevices enumera un conjunto de dispositivos.

Cuando implementar:

No se debe implementar esta interfaz. La implementación de Microsoft estándar proporciona una funcionalidad completa.

Cuándo utilizar:

Utilice esta interfaz para revisar los dispositivos devueltos por las búsquedas anteriores.

Nota: para C + +, se recuperan todas las propiedades por un método que se llama anteponiendo get_ al nombre de la propiedad. Por ejemplo, si el nombre de la propiedad es SomeState, entonces get_SomeState recupera la propiedad.

Miembros:

La interfaz de IUPnPDevices hereda de la interfaz IDispatch. IUPnPDevices también define los siguientes tipos de miembros:

Métodos:

Propiedades
Métodos

La interfaz IUPnPDevices define el método siguiente.

Método Descripción
Esta interfaz no tiene métodos. N / A

Propiedades
La interfaz IUPnPDevices define las siguientes propiedades.

Descripción de la Propiedad de acceso de tipo
_NewEnum
Enumerador de interfaz para la colección
Count
Número de dispositivos de la colección..

Item
Una interfaz IUPnPDevice de la colección.

Requerimientos:

Mínimos admitidos cliente
Windows XP
Mínimos admitidos servidor
Ninguno apoyo
Encabezamiento
UPnP.h
DLL
UPnP.dll

I. IUPnPHttpHeaderControl Interfaz

La interfaz permite a la persona que llama IUPnPHttpHeaderControl para especificar cabeceras HTTP adicionales enviados en peticiones HTTP a un dispositivo.

Cuando implementar:

Microsoft proporciona una implementación completa de esta interfaz.

Miembros:

La interfaz de IUPnPHttpHeaderControl hereda de la interfaz IUnknown. IUPnPHttpHeaderControl también define los siguientes tipos de miembros:

Métodos:

Métodos
La interfaz IUPnPHttpHeaderControl define el método siguiente.

Método Descripción
AddRequestHeaders Agrega el encabezado HTTP suministrado a una petición HTTP.

Comentarios:

Esta interfaz se obtiene llamando a QueryInterface en el mismo objeto que proporciona una implementación de la IUPnPDeviceFinder o interfaces

IUPnPDescriptionDocument, tras lo cual se puede llamar AddRequestHeaders en él.

Requerimientos:

Mínimos admitidos cliente
Windows 7
Mínimos admitidos servidor
Ninguno apoyo
Encabezamiento
UPnP.h
DLL
UPnP.dll

J. IUPnPService Interfaz

La interfaz IUPnPService permite a una aplicación variables de estado de consulta y ejecutar acciones en una instancia de un servicio.

Cuando implementar:

No se debe implementar esta interfaz. La implementación de Microsoft estándar proporciona una funcionalidad completa.

Cuándo utilizar:

Utilice esta interfaz para controlar un dispositivo.

Nota: para C + +, se recuperan todas las propiedades por un método que se llama

anteponiendo get_ al nombre de la propiedad. Por ejemplo, si el nombre de la propiedad es SomeState, entonces get_SomeState recupera la propiedad.

Miembros:

La interfaz de IUPnPService hereda de la interfaz IDispatch. IUPnPService también define los siguientes tipos de miembros:

Métodos
Propiedades
Métodos
La interfaz IUPnPService define los siguientes métodos.

Método Descripción
Devuelve QueryStateVariable el valor de la variable de estado especificado.

InvokeAction Invoca la acción especificada.

Registra una devolución de llamada AddCallback servicio.

Propiedades:

La interfaz IUPnPService define las siguientes propiedades.

Propiedad tipo de acceso Descripción
Id
Servicio de ID para el servicio
LastTransportStatus
de estado HTTP de la última solicitud enviada al servicio en el dispositivo. La solicitud es para invocar una acción o consultar el valor de una variable de estado

no evented.

ServiceTypeIdentifier
Servicio de identificador de tipo por el servicio

Requerimientos:

Mínimos admitidos cliente
Windows XP
Mínimos admitidos servidor
Ninguno apoyo
Encabezamiento
UPnP.h
DLL
UPnP.dll

K. IUPnPServiceCallback Interfaz

La interfaz IUPnPServiceCallback se utiliza para enviar notificaciones de eventos a los clientes de los objetos de servicio.

Cuando implementar:

Los desarrolladores deben implementar esta interfaz para aplicaciones C + + que se escriben para recibir notificaciones de eventos desde un servicio. Los desarrolladores que utilizan todos los lenguajes de programación, como Visual Basic, debe implementar una función de devolución de llamada personalizados para recibir devoluciones de llamada. Ver Registro de una devolución de llamada

para los ejemplos.

Cuándo utilizar:

El entorno UPnP hace referencia a esta devolución de llamada cuando se produce un evento.

Miembros:

La interfaz de IUPnPServiceCallback hereda de la interfaz IUnknown. IUPnPServiceCallback también define los siguientes tipos de miembros:

Métodos
Métodos
La interfaz IUPnPServiceCallback define los siguientes métodos.

Método Descripción
StateVariableChanged Método que indica que una variable de estado ha cambiado.

ServiceInstanceDied Método que indica que un servicio ya no será el envío de eventos.

Requerimientos:

Mínimos admitidos cliente
Windows XP
Mínimos admitidos servidor
Ninguno apoyo

Encabezamiento
UPnP.h
DLL
UPnP.dll

L. IUPnPServices Interfaz

La interfaz IUPnPServices enumera un conjunto de servicios.

Cuando implementar:

No se debe implementar esta interfaz. La implementación de Microsoft estándar proporciona una funcionalidad completa.

Cuándo utilizar:

Utilice esta interfaz para enumerar la lista de los servicios asociados a un dispositivo.

Tenga en cuenta que el uso de estas propiedades hacer que el entorno UPnP en contacto con el servicio a través de la red. Si el servicio no existe, la aplicación se bloquea hasta que la solicitud de conexión falla. Se recomienda que las aplicaciones utilicen la propiedad del artículo en la mayoría de los casos, y la reserva de la enumeración de casos cuando una aplicación se está iniciando.

Nota: para C + +, se recuperan todas las propiedades por un método que se llama anteponiendo get_ al nombre de la propiedad. Por ejemplo, si el nombre de la propiedad es SomeState, recupera get_SomeState la propiedad.

Miembros:

La interfaz de IUPnPServices hereda de la interfaz IDispatch. IUPnPServices también define los siguientes tipos de miembros:

Métodos
Propiedades
Métodos
La interfaz IUPnPServices define el método siguiente.

Método Descripción
Esta interfaz no tiene métodos. N / A

Propiedades:

La interfaz IUPnPServices define las siguientes propiedades.

Descripción de la Propiedad de acceso de tipo

_NewEnum
Enumerador de interfaz para la colección.
Count
Número de servicios de la colección.

Item
Una interfaz IUPnPService en la colección

Requerimientos:

Mínimos admitidos cliente

Windows XP

Mínimos admitidos servidor

Ninguno apoyo

Encabezamiento

UPnP.h

DLL

UPnP.dll

ANEXO II: DIAGRAMAS UML

Este anexo es un breve tutorial sobre UML para la descripción de los diagramas manejados por este lenguaje y que han sido utilizados en la elaboración del presente proyecto.

Introducción

El Lenguaje de Modelamiento Unificado (UML: *Unified Modeling Language*) es un lenguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de software. [UMLTUTORIAL]
A continuación se describe el formato de los tres tipos de diagramas UML que se pueden encontrar en varios apartados de la memoria:

- Diagrama de Casos de Uso.
- Diagrama de Clases.
- Diagrama de Secuencias.

1. Diagrama de Casos de Uso

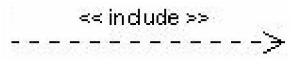
Este diagrama pretende representar de forma gráfica y sencilla la funcionalidad de un sistema y su interacción con los actores que intervienen. A continuación se describen las distintas entidades que figuran en el diagrama, así como sus posibles relaciones:

- Actores: son entidades externas al sistema pero que demandan o provocan una funcionalidad de este. Pueden ser tanto personas como organizaciones o como otros sistemas.

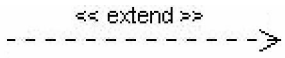
- Casos de uso: expresa una unidad coherente de funcionalidad del sistema. Es decir, algo que el sistema realiza.

Relaciones entre los casos de uso:

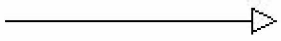
Inclusión («include»): una instancia del Caso de Uso origen incluye también el comportamiento descrito por el Caso de Uso destino.



Extensión («extend»): El caso de uso origen extiende el comportamiento del caso de uso destino. Es una especificación de la funcionalidad que describe el caso de destino.



Herencia: El caso de uso origen hereda la especificación del caso de uso destino y posiblemente la modifica y/o amplía.



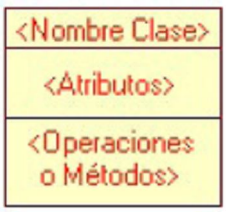
2. Diagrama de Clases

2.1 Introducción

Un diagrama de clases sirve para visualizar las relaciones entre las clases que involucran el sistema, las cuales pueden ser asociativas, de herencia, de uso y de contenimiento.
Un diagrama de clases está compuesto por los siguientes elementos:
* Clases: atributos, métodos y visibilidad.
* Relaciones: Herencia, Composición, Agregación, Asociación y Uso.

2.2 Clase

Es la unidad básica que encapsula toda la información de un Objeto (un objeto es una instancia de una clase). A través de ella podemos modelar el entorno en estudio (una Casa, un Auto, una Cuenta Corriente, etc.).
En UML, una clase es representada por un rectángulo que posee tres divisiones:



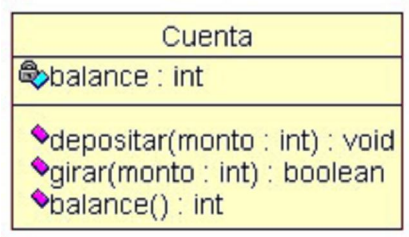
En donde:

* **Superior**: Contiene el nombre de la Clase.

* **Intermedio**: Contiene los atributos (o variables de instancia) que caracterizan a la Clase (pueden ser private, protected o public).

* **Inferior**: Contiene los métodos u operaciones, los cuales son la forma como interactúa el objeto con su entorno (dependiendo de la visibilidad: private, protected o public).

Ejemplo:
Una Cuenta Corriente que posee como característica:
- Balance
Puede realizar las operaciones de:
- Depositar
- Girar
- Balance
El diseño asociado es:



Atributos:

Los atributos o características de una Clase pueden ser de tres tipos, los que definen el grado de comunicación y visibilidad de ellos con el entorno, estos son:

* **Public** (+): Indica que el atributo será visible tanto dentro como fuera de la clase, es decir, es accesible desde todos lados.

* **Private** (-): Indica que el atributo sólo será accesible desde dentro de la clase (sólo sus métodos lo pueden accesar).

* **Protected** (#): Indica que el atributo no será accesible desde fuera de la clase, pero si podrá ser accesado por métodos de la clase además de las subclases que se deriven (ver herencia).

Métodos:

Los métodos u operaciones de una clase son la forma en la cual ésta interactúa con su entorno, éstos pueden tener las características:

* **Public** (+): Indica que el método será visible tanto dentro como fuera de la clase, es decir, es accesible desde todos lados.

* **Private** (-): Indica que el método sólo será accesible desde dentro de la clase (sólo otros métodos de la clase lo pueden accesar).

* **Protected** (#): Indica que el método no será accesible desde fuera de la clase, pero si podrá serlo por métodos de la clase además de métodos de las subclases que se deriven.

2.3 Relaciones entre Clases:

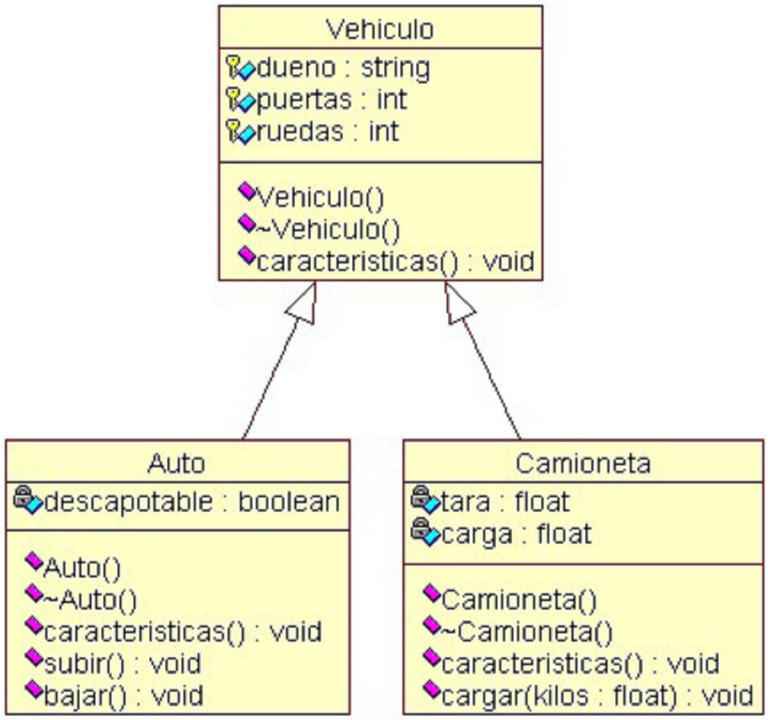
Ahora ya definido el concepto de Clase, es necesario explicar cómo se pueden interrelacionar dos o más clases (cada uno con características y objetivos diferentes).

Antes es necesario explicar el concepto de cardinalidad de relaciones: En UML, la cardinalidad de las relaciones indica el grado y nivel de dependencia, se anotan en cada extremo de la relación y éstas pueden ser:

* **uno o muchos**: 1..* (1..n)
* **0 o muchos**: 0..* (0..n)
* **número fijo**: m (m denota el número).

Herencia (Especialización/Generalización): 

Indica que una subclase hereda los métodos y atributos especificados por una Super Clase, por ende la Subclase además de poseer sus propios métodos y atributos, poseerá las características y atributos visibles de la Super Clase (public y protected), ejemplo:



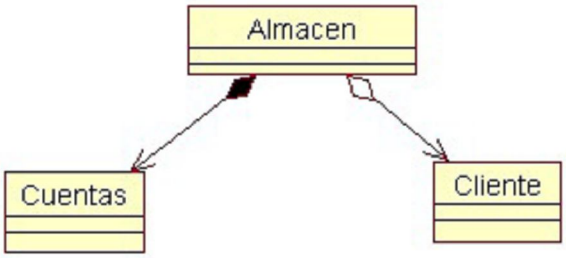
En la figura se especifica que Auto y Camión heredan de Vehículo, es decir, Auto posee las Características de Vehículo (Precio, VelMax, etc) además posee algo particular que es Descapotable, en cambio Camión también hereda las características de Vehiculo (Precio, VelMax, etc.) pero posee como particularidad propia Acoplado, Tara y Carga.
Cabe destacar que fuera de este entorno, lo único "visible" es el método Características aplicable a instancias de Vehículo, Auto y Camión, pues tiene definición pública, en cambio atributos como Descapotable no son visibles por ser privados.

Agregación: 

Para modelar objetos complejos, no bastan los tipos de datos básicos que proveen los lenguajes: enteros, reales y secuencias de caracteres. Cuando se requiere componer objetos que son instancias de clases definidas por el desarrollador de la aplicación, tenemos dos posibilidades:

* **Por Valor**: Es un tipo de relación estática, en donde el tiempo de vida del objeto incluido está condicionado por el tiempo de vida del que lo incluye. Este tipo de relación es comúnmente llamada **Composición** (el Objeto base se construye a partir del objeto incluido, es decir, es "parte/todo").

* **Por Referencia**: Es un tipo de relación dinámica, en donde el tiempo de vida del objeto incluido es independiente del que lo incluye. Este tipo de relación es comúnmente llamada **Agregación** (el objeto base utiliza al incluido para su funcionamiento).
Un Ejemplo es el siguiente:



En donde se destaca que:

- Un Almacén posee Clientes y Cuentas (los rombos van en el objeto que posee las referencias).
- Cuando se destruye el Objeto Almacén también son destruidos los objetos Cuenta asociados, en cambio no son afectados los objetos Cliente asociados.
- La composición (por Valor) se destaca por un rombo relleno.
- La agregación (por Referencia) se destaca por un rombo transparente.

La flecha en este tipo de relación indica la navegabilidad del objeto referenciado. Cuando no existe este tipo de particularidad la flecha se elimina.

Asociación: 

La relación entre clases conocida como Asociación, permite asociar objetos que colaboran entre sí. Cabe destacar que no es una relación fuerte, es decir, el tiempo de vida de un objeto no depende del otro.

Ejemplo:

Un cliente puede tener asociadas muchas Órdenes de Compra, en cambio una orden de compra solo puede tener asociado un cliente.

Dependencia o Instanciación (uso):

Representa un tipo de relación muy particular, en la que una clase es instanciada (su instanciación es dependiente de otro objeto/clase). Se denota por una flecha punteada.

El uso más particular de este tipo de relación es para denotar la dependencia que tiene una clase de otra, como por ejemplo una aplicación grafica que instancia una ventana (la creación del Objeto Ventana está condicionado a la instanciación proveniente desde el objeto Aplicación):

Cabe destacar que el objeto creado (en este caso la Ventana gráfica) no se almacena dentro del objeto que lo crea (en este caso la Aplicación).

3. Diagrama de Secuencias

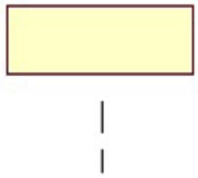
3.1 Introducción

El diagrama de interacción, representa la forma en como un Cliente (Actor) u Objetos (Clases) se comunican entre sí en petición a un evento. Esto implica recorrer toda la secuencia de llamadas, de donde se obtienen las responsabilidades claramente.

Dicho diagrama puede ser obtenido de dos partes, desde el Diagrama Estático de Clases o el de Casos de Uso (son diferentes).
Los componentes de un diagrama de interacción son:

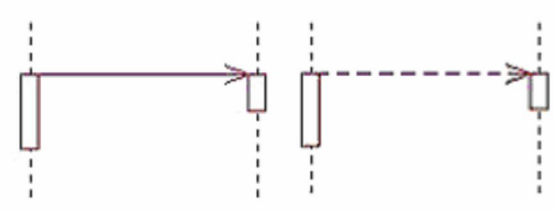
* Un Objeto o Actor.
* Mensaje de un objeto a otro objeto.
* Mensaje de un objeto a sí mismo.

3.2 Objeto/Actor:



El rectángulo representa una instancia de un Objeto en particular, y la línea punteada se utiliza para representar las llamadas a métodos del objeto de forma secuencial.

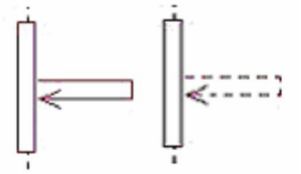
a. **Mensaje a Otro Objeto**:



Se representa por una flecha entre un objeto y otro, representa la llamada de un método (operación) de un objeto en particular. La línea discontinua indica el final de la ejecución del método. Esta línea aparece en el caso de que dentro del

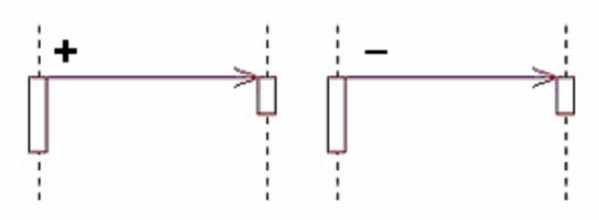
método se hagan llamadas a otros métodos, los cuales se muestran en el diagrama justo por debajo de su llamada y antes de ésta.

b. **Mensaje al Mismo Objeto**:



No solo se pueden realizar llamadas a métodos de objetos externos, también es posible visualizar llamadas a métodos desde el mismo objeto en estudio. Si el color de la línea es rojo, el método está definido como privado (sólo puede ser invocado dentro de la clase). De la misma manera que en el aparatado anterior, la línea continúa indica el final de la ejecución del método.

En la siguiente figura se muestran los símbolos "**+**" y "**-**". El primero indica que el método está comprimido, es decir, invoca a otros métodos del análisis pero que no son mostrados. El siguiente indica que el método esta expandido, por lo que se muestran los métodos que invoca justo debajo y antes del símbolo correspondiente de final de método.



ANEXO III: INSTRUCCIONES PARA LA INSTALACIÓN

En este apartado se informa los pasos necesarios para poner a funcionar la aplicación del punto de control.

1. Requerimientos del sistema:

A continuación se enumeran los requerimientos previos del sistema y en base a los cuales la aplicación ha sido desarrollada y probada:

- Hardware: PC con un mínimo de 256MB de RAM y procesador de 800MHz.
- Sistemas Operativos: plataformas Windows de 32 bits. Los sistemas operativos utilizados han sido Windows XP
- UPnP: debe estar habilitado UPnP en el sistema operativo y configurado el cortafuegos para que permita su uso.

2. Requerimientos de la aplicación:

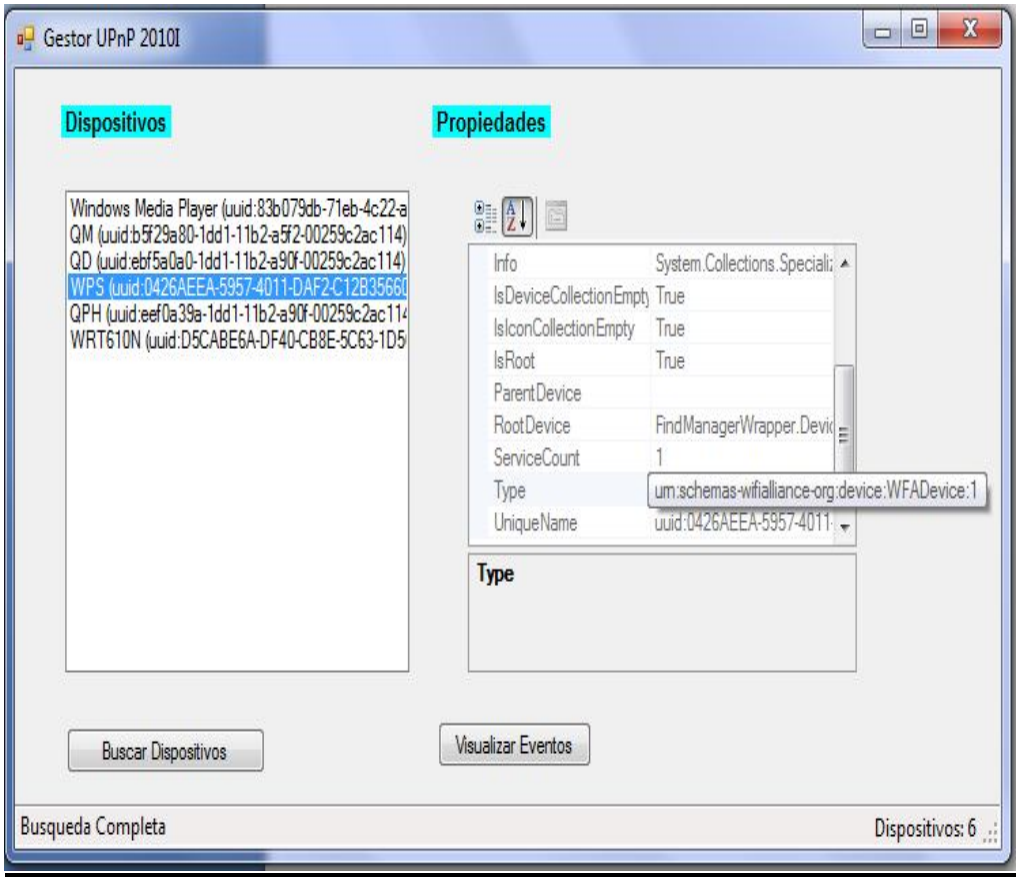
*.NET Framework 3.5 es necesario en el sistema operativo Windows XP, es utilizado por los programas realizados por los desarrolladores, pues brinda soluciones pre-codificadas para requerimientos comunes de los programas y gestiona la ejecución.

Uno de sus componentes son los formularios Windows Forms que fueron usados en este proyecto para la realización de la interfaz.

ANEXO IV: VERSIONES DEL SOFTWARE DESARROLLADAS

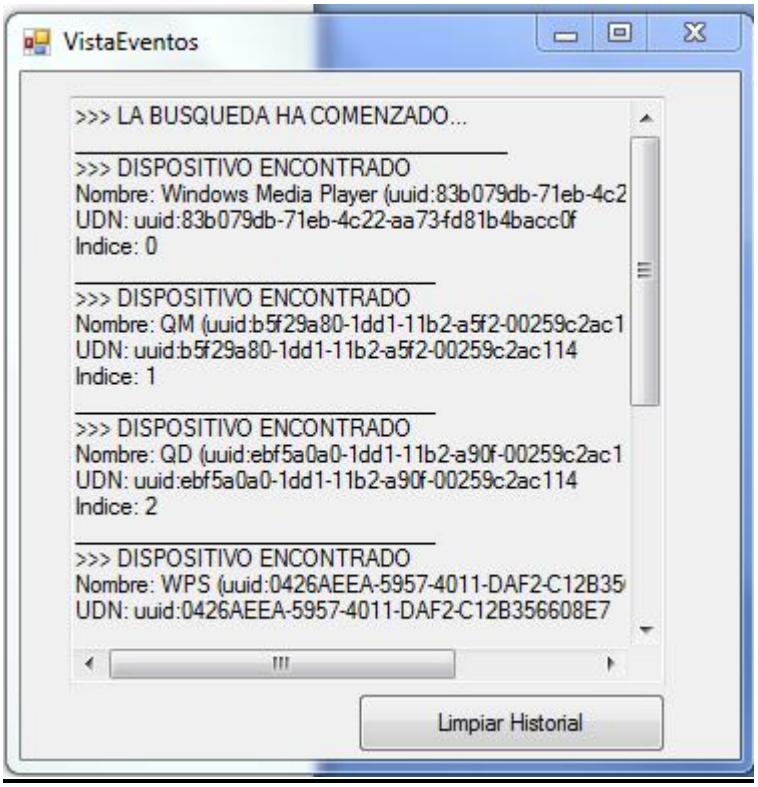
En el siguiente anexo veremos algunas imágenes de las versiones desarrolladas anteriormente.

Version GestorUPnP 2010I



En esta primera version se mostraba una lista de los dispositivos encontrados y se daba una breve descripcion de los mismos, ademas se daba la opcion de

observar el historial de eventos mediante el empleo de otra ventana que registraba cada uno de estos.

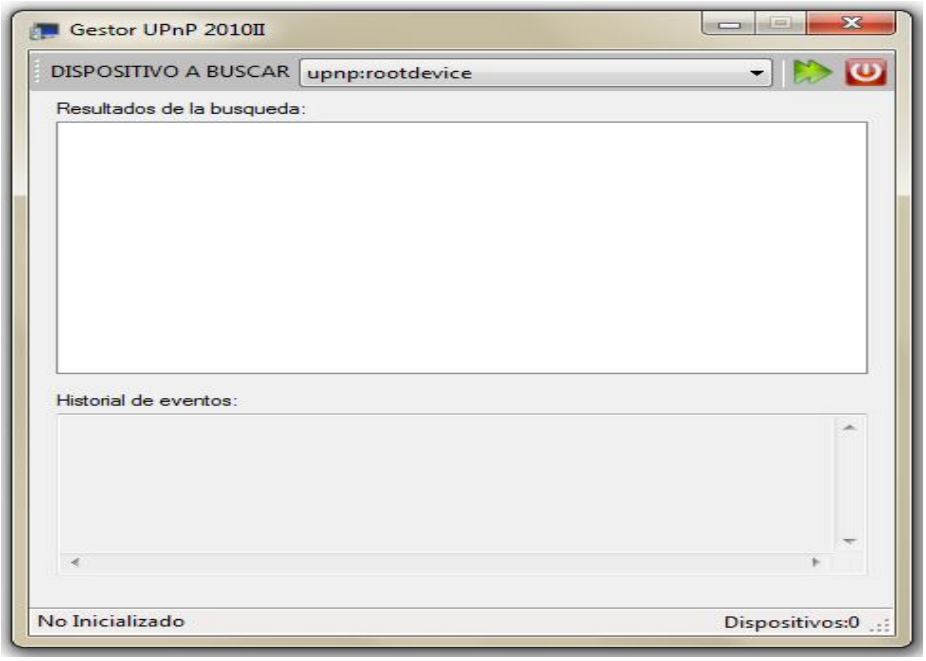


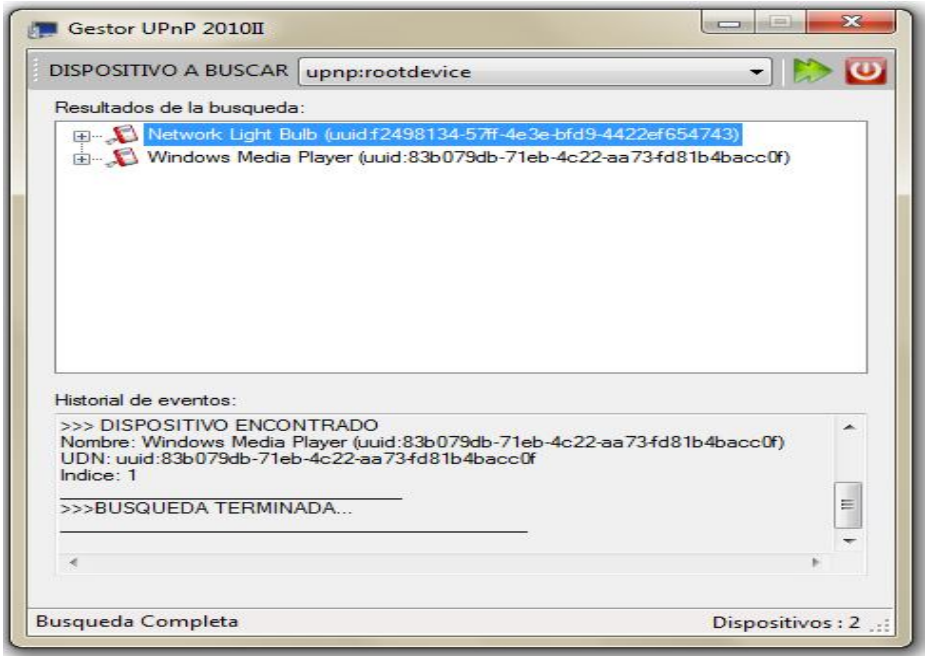
Version GestorUPnP 2010II

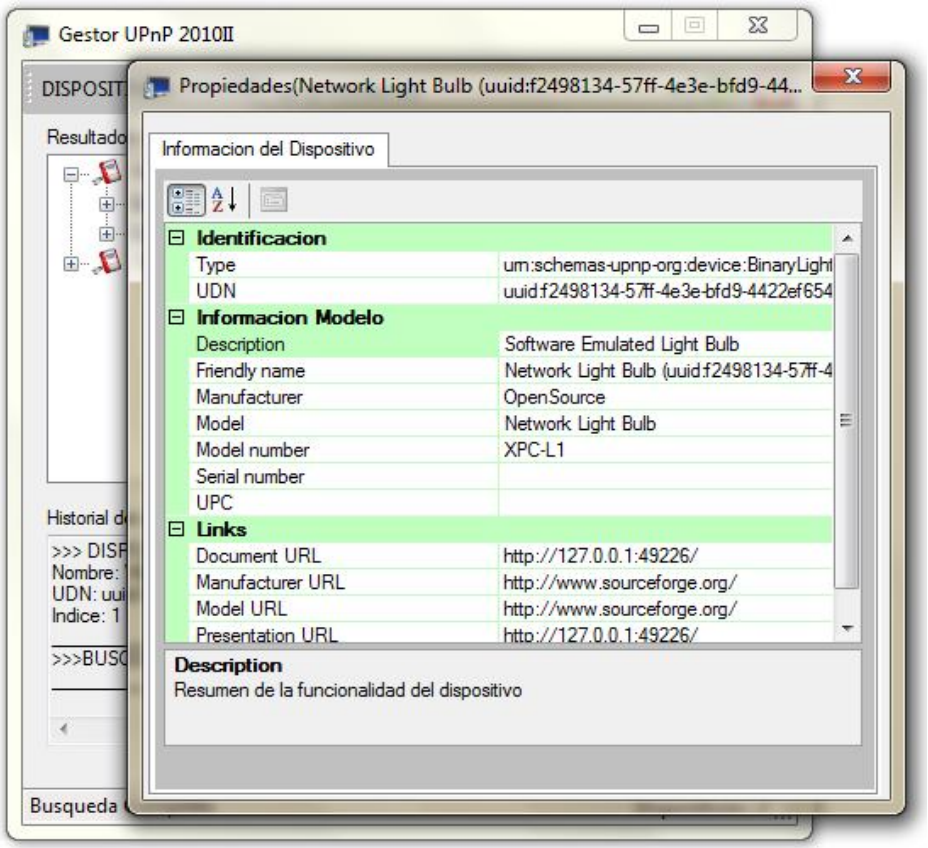
En esta versión podemos ver que es más completa, se puede observar un cambio en la presentación, el historial de eventos va incluido en la misma interfaz, aparte se da una información más completa de lo relacionado con la descripción del dispositivo, también se da la opción de poder seleccionar el tipo de dispositivo

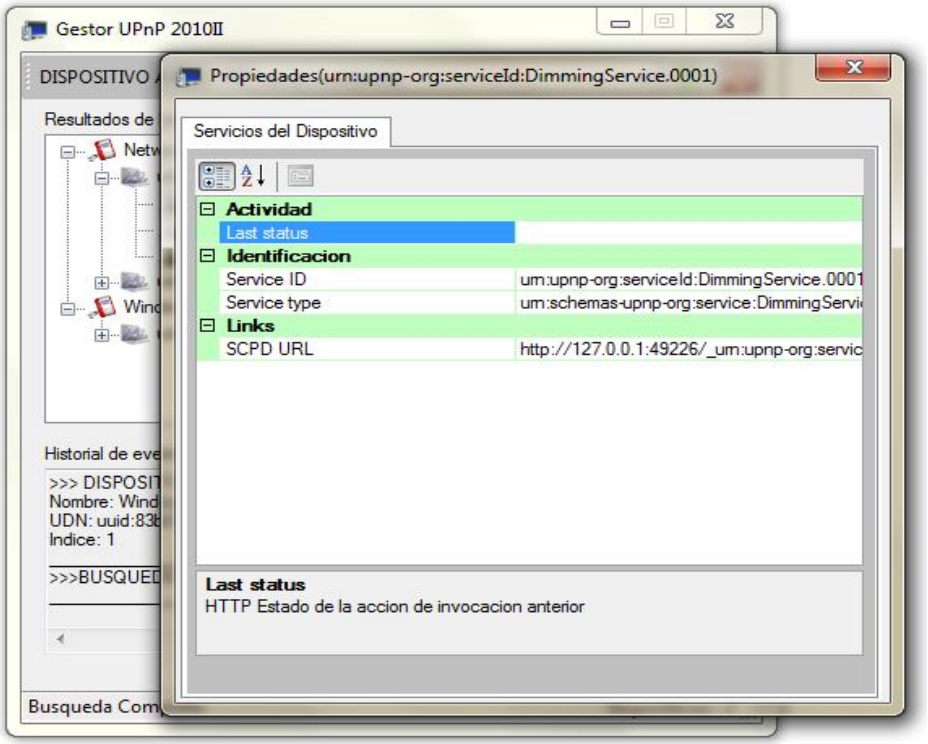
para buscar, junto con dos botones, uno para iniciar la búsqueda y otro para detenerla.

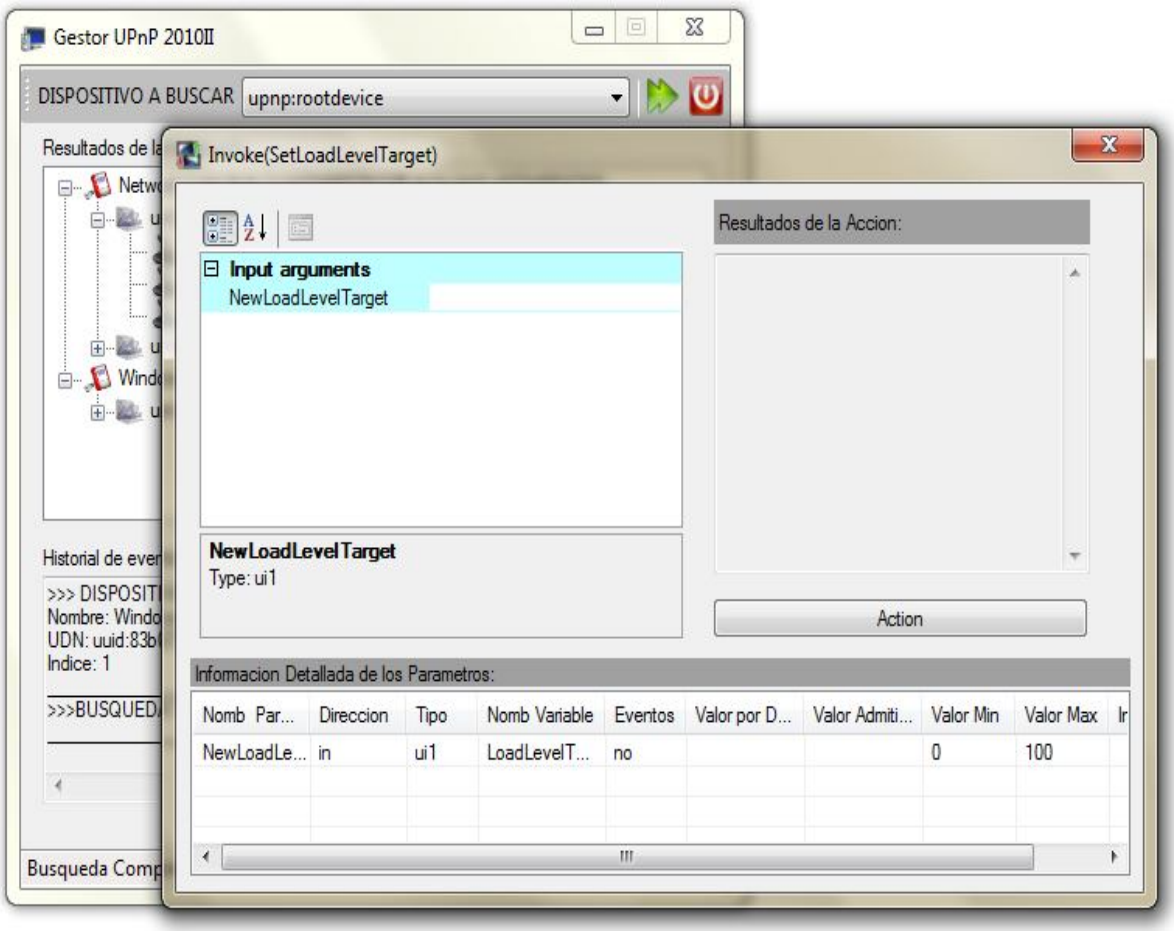
Además se accede a una información mucho más detallada sobre los dispositivos encontrados en la red, junto con la opción de ejecutar acciones sobre estos.





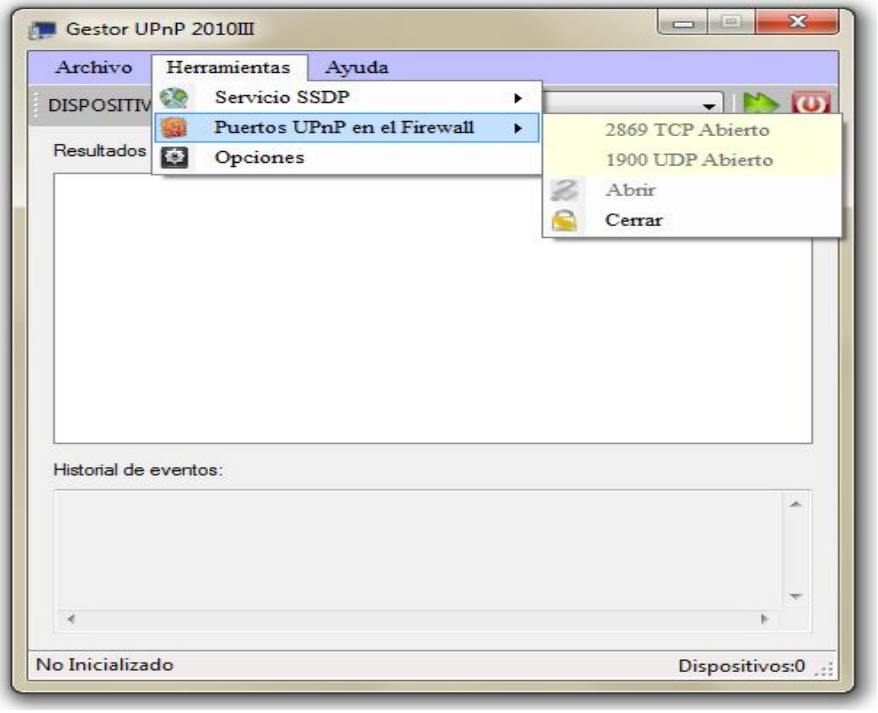

Gestor UPnP 2010II
DISPOSIT
Resultado
Historial d
>>> DISP
Nombre:
UDN: uui
Indice: 1
>>>BUSC
Busqueda
Propiedades(Network Light Bulb (uuid:f2498134-57ff-4e3e-bfd9-44...
Informacion del Dispositivo
Identificacion
Type
um:schemas-upnp-org:device:BinaryLight
UDN
uuid:f2498134-57ff-4e3e-bfd9-4422ef654
Informacion Modelo
Description
Software Emulated Light Bulb
Friendly name
Network Light Bulb (uuid:f2498134-57ff-4
Manufacturer
OpenSource
Model
Network Light Bulb
Model number
XPC-L1
Serial number
UPC
Links
Document URL
http://127.0.0.1:49226/
Manufacturer URL
http://www.sourceforge.org/
Model URL
http://www.sourceforge.org/
Presentation URL
http://127.0.0.1:49226/
Description
Resumen de la funcionalidad del dispositivo

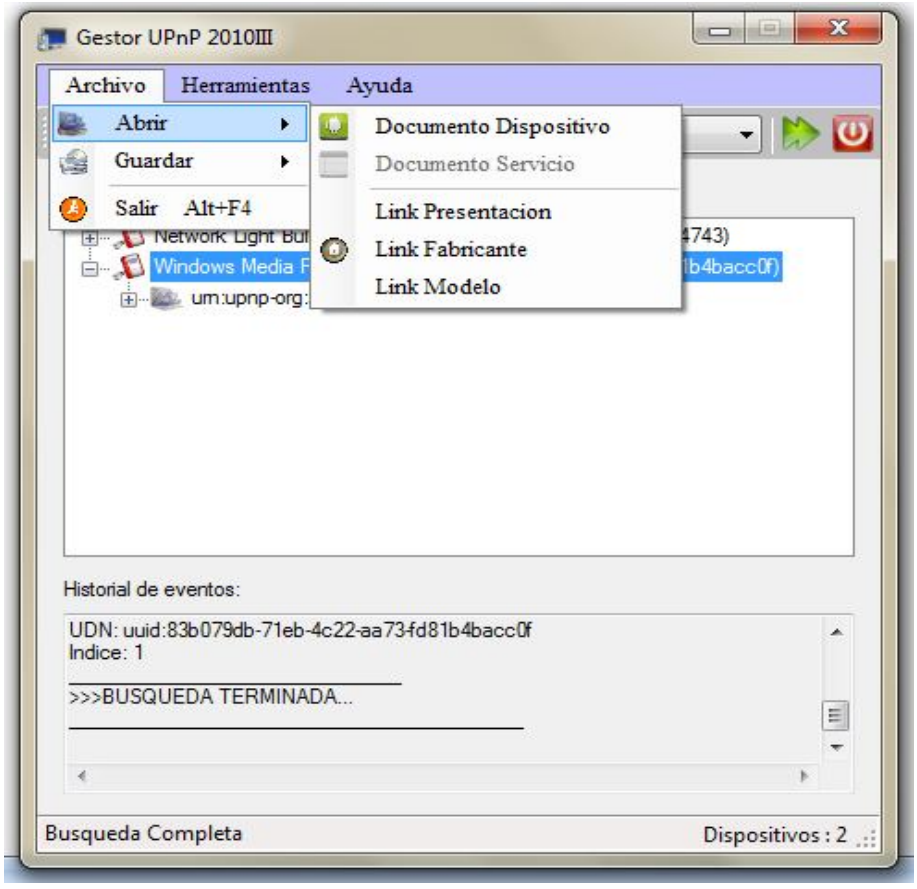

Gestor UPnP 2010II
DISPOSITIVO
Resultados de
Historial de eve
>>> DISPOSIT
Nombre: Wind
UDN: uuid:83
Indice: 1
>>>BUSQUED
Busqueda Com
Propiedades(urn:upnp-org:serviceId:DimmingService.0001)
Servicios del Dispositivo
Actividad
Last status
Identificacion
Service ID
urn:upnp-org:serviceId:DimmingService.0001
Service type
urn:schemas-upnp-org:service:DimmingServi
Links
SCPD URL
http://127.0.0.1:49226/_urn:upnp-org:servic
Last status
HTTP Estado de la accion de invocacion anterior

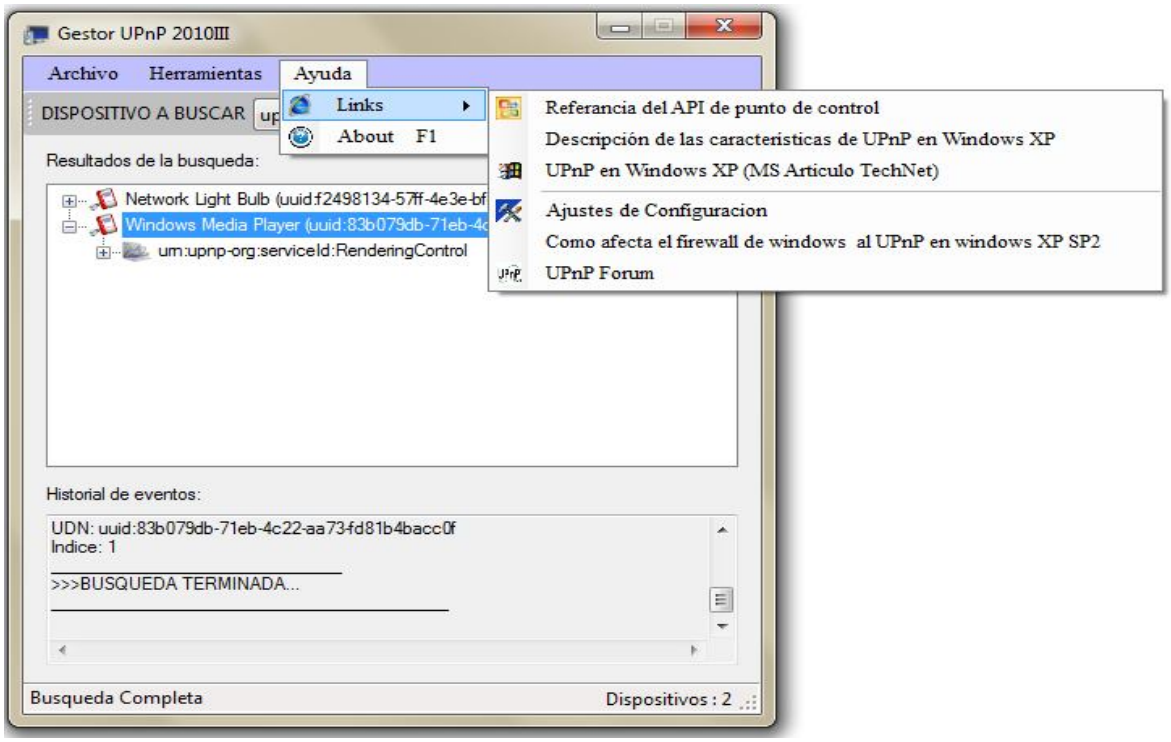


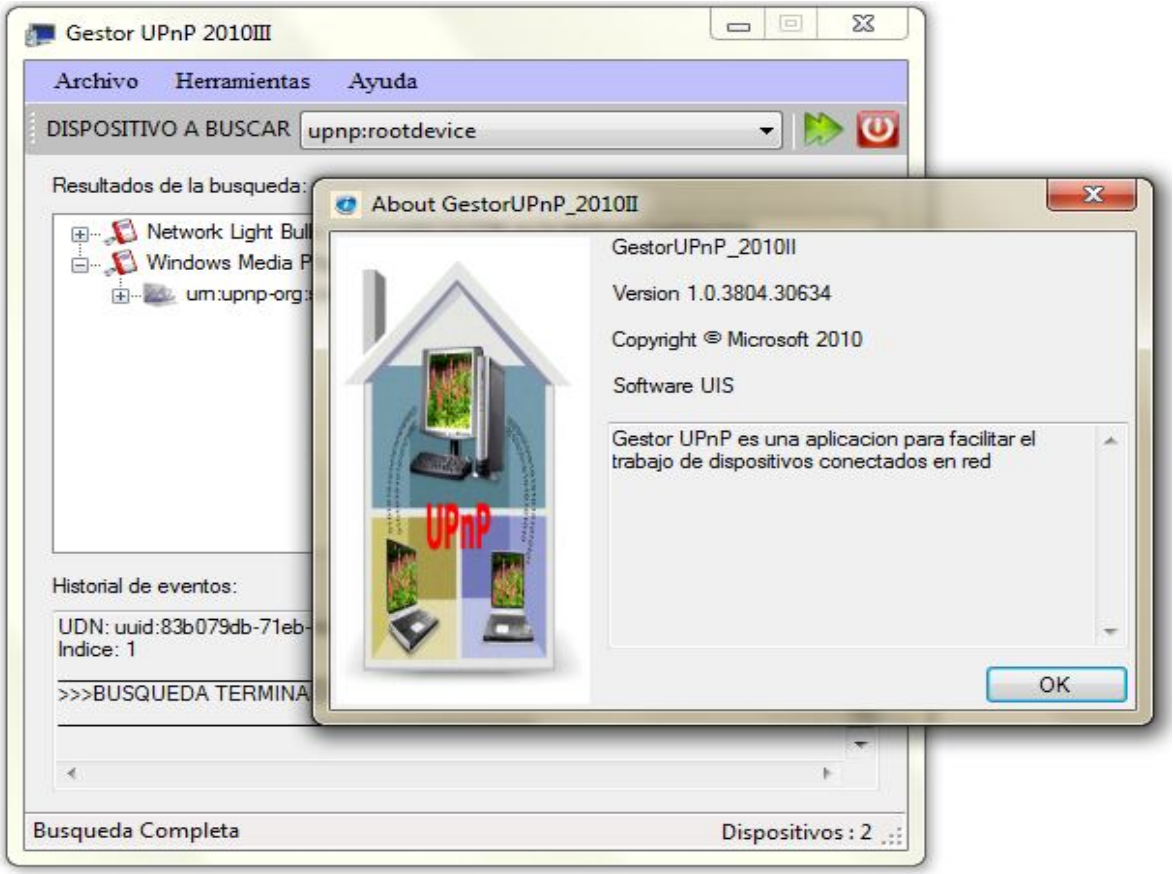
Version GestorUPnP 2010III

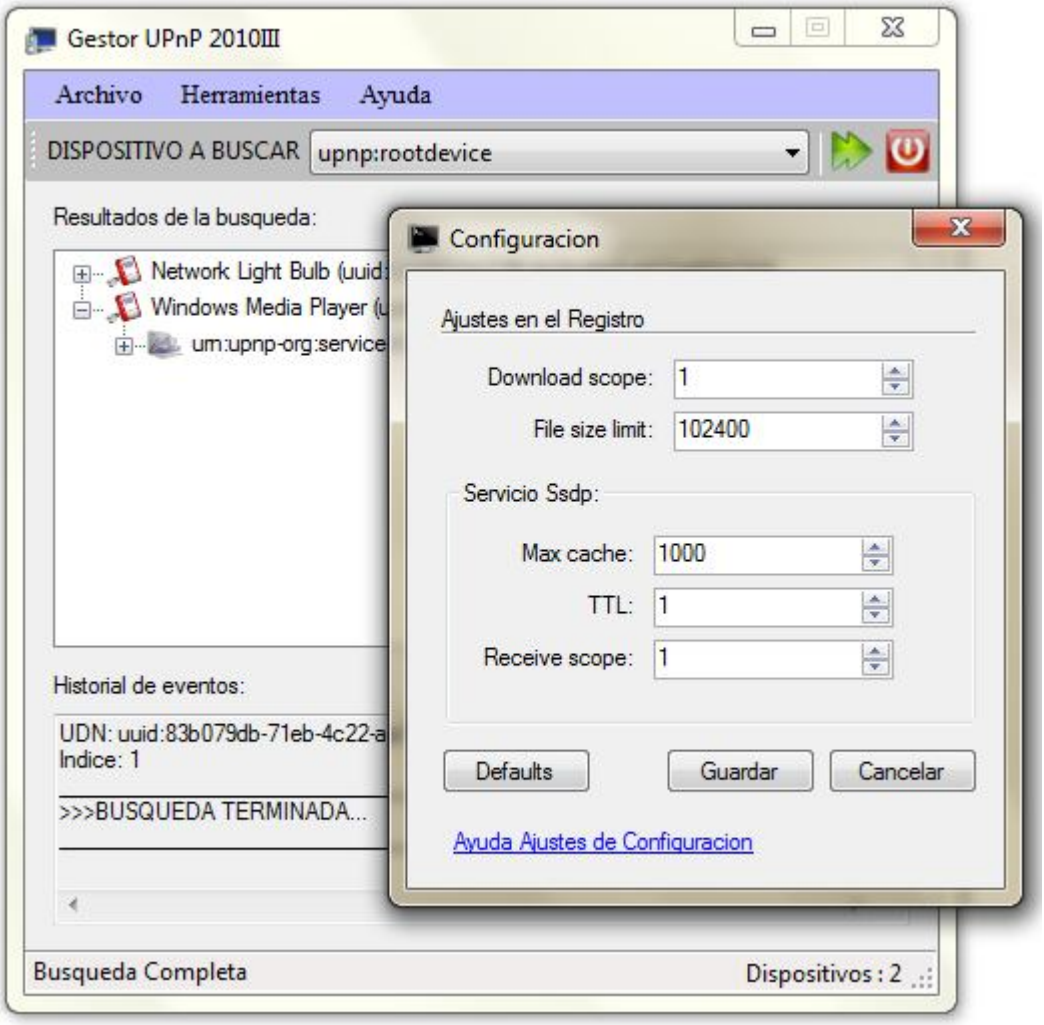
En esta tercera versión del programa se ha cambiado un poco su aspecto con otros colores y se han agregado otras funciones como son la activación de los puertos TCP y UDP, al igual se han agregado algunos links para tener enlace a mayor información relativa al UPnP , a los dispositivos y al fórum UPnP.


Gestor UPnP 2010III
Archivo
Herramientas
Ayuda
Servicio SSDP
Puertos UPnP en el Firewall
Opciones
2869 TCP Abierto
1900 UDP Abierto
Abrir
Cerrar
Resultados
Historial de eventos:
No Inicializado
Dispositivos:0


Gestor UPnP 2010III
Archivo
Herramientas
Ayuda
Abrir
Guardar
Salir Alt+F4
Documento Dispositivo
Documento Servicio
Link Presentacion
Link Fabricante
Link Modelo
Windows Media F
um:upnp-org:
Historial de eventos:
UDN: uuid:83b079db-71eb-4c22-aa73-fd81b4bacc0f
Indice: 1
>>>BUSQUEDA TERMINADA...
Busqueda Completa
Dispositivos : 2


Gestor UPnP 2010III
Archivo
Herramientas
Ayuda
DISPOSITIVO A BUSCAR
Links
About F1
Referancia del API de punto de control
Descripción de las caracteristicas de UPnP en Windows XP
UPnP en Windows XP (MS Articulo TechNet)
Ajustes de Configuracion
Como afecta el firewall de windows al UPnP en windows XP SP2
UPnP Forum
Resultados de la busqueda:
Network Light Bulb (uuid:f2498134-57ff-4e3e-bf
Windows Media Player (uuid:83b079db-71eb-4
um:upnp-org:serviceId:RenderingControl
Historial de eventos:
UDN: uuid:83b079db-71eb-4c22-aa73-fd81b4bacc0f
Indice: 1
>>>BUSQUEDA TERMINADA...
Busqueda Completa
Dispositivos : 2


Gestor UPnP 2010III
Archivo
Herramientas
Ayuda
DISPOSITIVO A BUSCAR
upnp:rootdevice
Resultados de la busqueda:
Network Light Bul
Windows Media P
um:upnp-org:
Historial de eventos:
UDN: uuid:83b079db-71eb-
Indice: 1
>>>BUSQUEDA TERMINA
Busqueda Completa
Dispositivos : 2
About GestorUPnP_2010II
GestorUPnP_2010II
Version 1.0.3804.30634
Copyright © Microsoft 2010
Software UIS
Gestor UPnP es una aplicacion para facilitar el trabajo de dispositivos conectados en red
UPnP
OK



A continuación se anexa un pantallazo de la actualización mas reciente de la aplicación, en la cual se ha cambiado un poco el diseño, pero lo más importante es que se ha agregado la funcionalidad de activar o desactivar el modo colaboración, que precisamente permite la colaboración entre dispositivos tipo switch y tipo bombilla.


Gestor UPnP 2010III
Archivo
Herramientas
Ayuda
DISPOSITIVO A BUSCAR
upnp:rootdevice
GESTOR UPnP 2010III
Windows Media Player (uuid:9e50f60b-37c9-428c-a2e9-3f77b84710af)
Bombilla Binaria.NET S27 (uuid:87206d34-8308-4123-ac84-777588f68f77)
ENCENDER
APAGAR
REGULAR...
Historial de eventos:
Activar Modo Colaboracion
>>> LA BUSQUEDA HA COMENZADO...
>>> DISPOSITIVO ENCONTRADO
Indice: 1
Nombre: Windows Media Player (uuid:9e50f60b-37c9-428c-a2e9-3f77b84710af)
UDN: uuid:9e50f60b-37c9-428c-a2e9-3f77b84710af
>>> DISPOSITIVO ENCONTRADO
Indice: 2
Nombre: Bombilla Binaria.NET S27 (uuid:87206d34-8308-4123-ac84-777588f68f77)
UDN: uuid:87206d34-8308-4123-ac84-777588f68f77
>>>BUSQUEDA TERMINADA...
Limpiar Historial
Busqueda Completa
Dispositivos : 2