# Plataforma de alto nivel para el desarrollo de sensores y actuadores de bajo coste en entornos inteligentes

C. Martín, D. Villa, O. Aceña, F.J. Villanueva, F. Moya, J.C. López

Laboratorio ARCO, Escuela Superior de Informática, Universidad de Castilla-La Mancha,
Paseo de la Universidad 4, 13071, Ciudad Real
{cleto.martin,david.villa,oscar.acena,felix.villanueva,
francisco.moya,juancarlos.lopez}@uclm.es
http://arco.esi.uclm.es

**Resumen** En este artículo se presenta la plataforma IcePick, un entorno de desarrollo software que facilita la creación de aplicaciones en redes de sensores y actuadores para infraestructuras y entornos inteligentes. Su principal ventaja es que la programación de los dispositivos de bajo coste se realiza desde un alto nivel de abstracción, agilizando el proceso de desarrollo. Además, la instalación de los dispositivos es transparente en un entorno inteligente ya desplegado.

**Keywords:** redes de sensores y actuadores, middleware, metodología de desarrollo

## 1. Introducción

Actualmente, las infraestructuras inteligentes presentan numerosos retos de investigación en diferentes campos de la computación. Uno de ellos son las redes de sensores y actuadores (SAN). Los sistemas inteligentes necesitan conocer diferentes magnitudes del entorno que les rodea tales como temperatura, presión, etc. En base a este tipo de datos pueden tomar decisiones y modificar el estado del entorno a través de actuadores como interruptores, válvulas, etc. Así, la gestión inteligente de infraestructuras requiere desplegar sensores y actuadores para que sean la interfaz con el mundo real.

Desde el punto de vista de la programación de este tipo de redes se presentan varios retos. En primer lugar, es necesario que las infraestructuras inteligentes sean *escalables*. Incluir nuevos dispositivos sensores o actuadores en un sistema ya desplegado puede ser una tarea complicada debido, principalmente, a la *heterogeneidad* de estos dispositivos. Por ejemplo, un nodo sensor de luz podría requerir adaptaciones específicas para su integración con el resto del sistema debido a que pertenece a otro fabricante, o funciona con otro sistema operativo o por cualquier otro motivo. Estos adaptadores complican tanto la programación del dispositivo (aumentando los costes asociados) así como el propio entorno inteligente, que necesita soportar servicios de traducción para interactuar con las SAN desde otras partes del sistema.

Desde el punto de la infraestructura inteligente, los sensores y actuadores tienen que hacer tareas sencillas tales como medición de una magnitud, envío de medidas, responder ante eventos simples, etc. Para la mayoría de las aplicaciones es suficiente el uso de dispositivos basados en microcontroladores de 8 bits y con pocos KBytes

de memoria, reduciendo en gran medida el coste de la instalación. Sin embargo, el inconveniente principal de este tipo de dispositivos es que las herramientas y lenguajes de programación ofrecen un *nivel de abstracción bajo*. Es por ello que su diseño precisa de personal más especializado, siendo costosos de desarrollar y de mantener. Cualquier cambio en la infraestructura requiere mucho tiempo de desarrollo y depuración en estos dispositivos.

Este trabajo describe una plataforma de desarrollo de dispositivos de bajo coste llamada IcePick. Para el programador ofrece un alto nivel de abstracción y permite construir SAN con nodos heterogéneos de muy bajo coste directamente integrables en un sistema ya desplegado.

En la sección 2 se muestra el trabajo relacionado. En la sección 3 se describen los componentes de IcePick y en la sección 4 se presentan los resultados de diferentes aplicaciones prototipo. Finalmente, en la sección 5 se exponen las conclusiones.

## 2. Trabajo relacionado

Existen en el mercado algunas soluciones que actualmente se utilizan para la programación de dispositivos sensores y actuadores. Waspmote es un producto creado por la empresa Libelium orientado a las redes de sensores. En estos dispositivos se pueden montar placas con sensores y actuadores y, junto con el producto, se proporcionan entornos de programación. El lenguaje de programación es C++ simplificado, restringiéndolo a la API[1] proporcionada por Libelium. Sin embargo, los entornos y librerías proporcionadas con Waspmote son exclusivamente para esta plataforma. De hecho, la biblioteca proporcionada no es genérica y depende en gran medida de la plataforma hardware de Waspmote.

Otros dispositivos reseñables son la gama Sun SPOT[2] y Lego Mindstorms que tienen implementaciones de la máquina virtual de Java (JVM), por lo que son programables en este lenguaje. Sin embargo, el tamaño de una implementación mínima de una JVM y sus requisitos hardware hacen inviable la carga de estas aplicaciones en dispositivos de baja gama como microcontroladores.

## 3. IcePick: metodología de desarrollo

La plataforma IcePick se basa en la filosofía de la *orientación a objetos distribuida*. Además de proporcionar un alto nivel de abstracción, la programación orientada a objetos distribuida aísla el acceso a la red de comunicaciones de forma que el programador no tenga en cuenta ese tipo de cuestiones. Los *middlewares* orientados a objetos (MOO) tales como CORBA[3] o Ice[4] proporcionan esta capa de abstracción de acceso a la red. Los nodos que conforman el sistema distribuido pueden ser heterogéneos (PCs, servidores, sensores, actuadores, etc.) ya que todos comparten el núcleo de comunicaciones.

[1] `http://www.libelium.com/development/waspmote`
[2] `http://uk.sun.com/specials/sunspot/spec.js`
[3] `http://www.corba.org`
[4] `http://www.zeroc.com`

En la terminología de los MOO, un *objeto distribuido* (OD) es una entidad que proporciona funcionalidad al resto del sistema. Por ejemplo, un sensor de temperatura se puede modelar como un OD que permite obtener un valor de temperatura a través de la invocación de determinado método. Para describir la funcionalidad y los modos de acceso a los servicios de un OD se utilizan *interfaces*. Dependiendo del middleware, las interfaces se escriben en un formato u otro (IDL para CORBA; Slice para Ice; etc.).

IcePick se basa en el concepto de picoObjeto [1,2], esto es, una implementación mínima de un OD que puede ser cargada en dispositivos de bajo coste y, a la vez, ser integrable automáticamente en un sistema distribuido ya desplegado. Su estructura es similar a muchos lenguajes interpretados como Java: un *bytecode* describe el programa y éste se ejecuta sobre una *máquina virtual* para una plataforma concreta. De esta forma, se consigue independencia de la arquitectura hardware en los nodos.

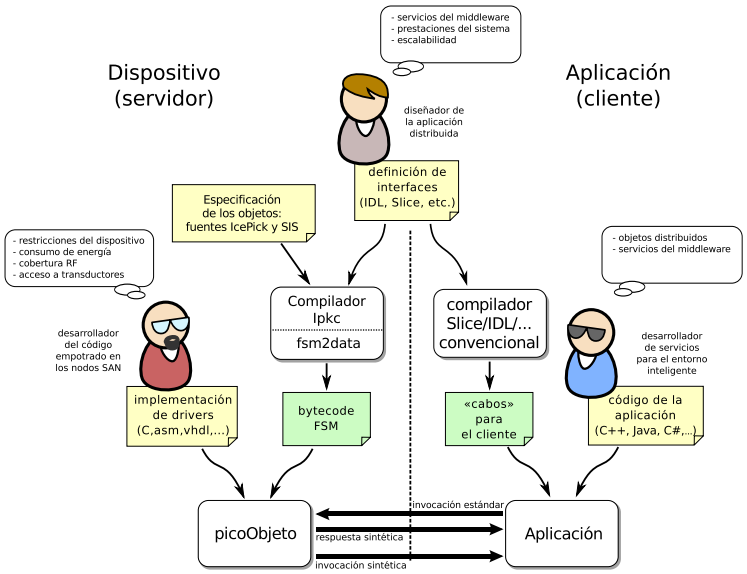


**Figura 1.** Esquema de la metodología de desarrollo con IcePick. La parte de la izquierda corresponde al desarrollo de un picoObjeto utilizando IcePick. La parte de la derecha es el desarrollo de una aplicación que interactúa con el picoObjeto. Nótese que partiendo de la descripción de las interfaces, el desarrollo puede hacerse en paralelo.

En definitiva, IcePick proporciona un conjunto de herramientas para crear picoObjetos automáticamente para diferentes plataformas a partir de una descripción abstracta y automatiza todo el proceso. Al soportar diferentes tecnologías y arquitecturas, la plataforma es fácilmente extensible.

En la figura 1 se muestra el flujo de desarrollo de una típica aplicación distribuida. Partiendo de las interfaces definidas por el diseñador, el diseñador de código empotrado define el comportamiento del picoObjeto utilizando el lenguaje de alto nivel de IcePick. El compilador *ipkc* traduce el comportamiento deseado a bytecode interpretable por el

picoObjeto y, además, se proporcionan drivers de acceso al hardware. Estos drivers deben ser implementados en lenguaje de bajo nivel, pero la plataforma IcePick permite su reutilización cuando sea necesario. Finalmente, la plataforma automatiza la carga de todo el software en el dispositivo final. Las aplicaciones clientes interactúan con el picoObjeto como si de un OD estándar se tratase, por lo que la integración es automática y sin necesidad de intermediarios.

## 4. Prototipos

IcePick ha sido utilizado en numerosos proyectos de investigación, la mayoría de ellos orientados a entornos inteligentes, como el proyecto Hesperia (CENIT Hesperia). En la Cuadro 1 se muestran los requisitos de máquina virtual para cada prototipo utilizando el middleware Ice.

**Cuadro 1.** Requisitos de máquina virtual de diferentes aplicaciones. La columna *adv* indica si el dispositivo puede anunciarse al sistema para ser localizado. Tamaños en *bytes*.

| **prototipo** | **adv** | **datos** | **código** | **total** | **RAM** |
|---|---|---|---|---|---|
| Pico mínimo | no | 14 | 89 | 103 | 18 |
| 2 interruptores | no | 68 | 374 | 442 | 19 |
| Cerradura electrónica | sí | 180 | 322 | 449 | 19 |
| Display electrónico | no | 71 | 292 | 363 | 19 |
| 3 válvulas, 3 sen. luz, 1 sen. presencia | sí | 209 | 846 | 1 055 | 22 |
| Termostato configurable | no | 250 | 626 | 876 | 20 |

A estas cifras se deben sumar los requisitos de la implementación para la máquina virtual que van desde 1 824 KBytes para microcontroladores PIC hasta 5 199 KBytes para dispositivos que utilicen algún sistema operativo (p.ej. TinyOS).

## 5. Conclusiones

IcePick proporciona una forma más rápida y abstracta de desarrollar objetos distribuidos para dispositivos empotrados y simplifica el proceso de desarrollo reduciendo los costes asociados. Gracias al concepto de picoObjeto se pueden utilizar middlewares de propósito general como CORBA o Ice en las SAN donde los recursos de cómputo son muy escasos. De esta forma, se facilita la integración de este tipo de dispositivos en los entornos inteligentes. En `http://arco.esi.uclm.es/icepick` puede obtenerse la plataforma así como más información.

## Referencias

1. Moya, F., Villa, D., Villanueva, F.J., Rincón, F., Barba, J., López, J.C., *Embedding Standard Distributed Object-Oriented Middlewares in Wireless Sensor Networks*, Journal on Wireless Communications and Mobile Computing (WCMC), 1 Mar 2009.
2. Martín, C., Villa, D., Villanueva, F.J., Moya, F., Santofimia, M.J., López, J.C., *A development model supporting integrative object oriented middlewares for sensor and actuator networks*, International Conference on Wireless Networks, Julio 2010.