TESIS DOCTORAL

ARQUITECTURA ABIERTA PARA EL DESPLIEGUE DE LABORATORIOS REMOTOS SOBRE TECNOLOGÍAS DE DESARROLLO DE SISTEMAS EMBEBIDOS

UNIVERSIDAD DE DEUSTO

FACULTAD DE INGENIERÍA

Presentada por D. IGNACIO ANGULO MARTÍNEZ

dentro del Programa de Doctorado en INGENIERÍA INFORMÁTICA Y TELECOMUNICACIÓN

Dirigida por el Dr. D. JAVIER GARCÍA ZUBÍA

El Director

El Doctorando

BILBAO, JULIO DE 2015

A Elisa

'Son vanas y están plagadas de errores las ciencias que no han nacido del experimento, madre de toda certidumbre'

~Leonardo Da Vinci ~

Resumen

Los laboratorios remotos han demostrado ser un recurso educativo de alto nivel que mejora el proceso de enseñanza y aprendizaje, cumpliendo con los objetivos experimentales requeridos en los cursos de ciencias.

La continua evolución de las tecnologías utilizadas para el desarrollo de sistemas embebidos, con apariciones constantes de nuevos dispositivos que ofrecen mejor rendimiento y capacidad de interconexión, exige una actualización continua en los contenidos, herramientas y sistemas de desarrollo utilizados para su aprendizaje. La exigencia de un codiseño hardware/software en el desarrollo de sistemas embebidos convierte en fundamental la experimentación con distintas tecnologías en la formación en esta disciplina.

Existen numerosas implementaciones de laboratorios remotos que permiten la experimentación sobre diferentes tecnologías de sistemas embebidos. Sin embargo su diseño se basa en arquitecturas cerradas que responden a los requisitos específicos de una experimentación concreta. Esto dificulta el despliegue de laboratorios remotos por profesionales no involucrados en la investigación en laboratorios remotos, frenando el despegue definitivo de esta tecnología.

El trabajo de investigación incluido en esta tesis propone una arquitectura abierta para facilitar el despliegue de laboratorios remotos que proporcionen experimentación con tecnologías embebidas a través de Internet garantizando la sostenibilidad de los mismos.

Esta arquitectura es el resultado del análisis de los principales laboratorios remotos de referencia desplegados por la comunidad investigadora y para su validación se han llevado a cabo un número suficiente de implementaciones que permiten evaluar el cumplimiento de cada uno de los requisitos establecidos para proporcionar una eficaz experimentación con tecnologías embebidas. Los diferentes laboratorios remotos incluidos en estas implementaciones no consisten únicamente en pruebas de concepto para la validación de la arquitectura propuesta, sino que también han demostrado su validez como herramientas didácticas entre la comunidad universitaria.

Abstract

Remote laboratories have proven to be a high-level educational resource that enhances the teaching and learning process. They provide the experimentation requirements in science courses.

The technologies for the development of embedded systems are constantly evolving. New devices providing better performance and interconnection capacity require both the development tools and the learning methodology to be updated. The need of hardware / software co-design in the development of embedded systems becomes critical the experimentation with different technologies in training in this discipline.

There are numerous implementations of remote laboratories that allow experimentation on different technologies of embedded systems. However its design is based on closed architectures designed to meet the specific requirements of a particular experiment. This hinders the deployment of remote laboratories by professionals not involved in its research and slows the definitive launch of this technology.

The research included in this PhD dissertation proposes an open architecture for easy deployment of remote laboratories that provide experimentation with embedded systems through the Internet.

This architecture is the result of analysis of the main references of remote laboratories deployed by the research community. In order to validate the proposed architecture a sufficient number of implementations has been carried out. The requirements for the effective provision of experimentation with embedded technologies have been evaluated in every implementation. The remote laboratories included in these implementations are not just a proof of concept to validate the proposed architecture, but have also proved their worth as teaching tools in the university community.

Laburpena

Urrutiko laborategiak goi mailako hezkuntza-baliabideak direla frogatu dute. Laborategi hauek ikaskuntza eta irakaskuntza prozesua hobetzen dute, eta zientzia-kurtsoetan eskatutako helburu esperimentalak betetzen dute.

Landatutako sistemak eraikitzeko erabiltzen diren teknologietan dagoen etengabeko eboluzioak, non gero eta hobeagoak diren gailu berriak etengabe agertzen diren, eguneratze konstante bat eskatzen du edukietan edota erremintetan. Landatutako sistema bat egiteak hardware-software diseinu bat eraikitzea eskatzen du. Diseinu hibrido hau egiteko ezinbestekoa da hainbat teknologiekin esperimentatzea.

Hainbat inplementazio existitzen dira urrutiko laborategiak eraikitzeko, zeinak landatutako sistemetan ohikoak diren teknologia desberdinekin saiakerak egitea ahalbidetzen duten. Hala ere, haien diseinua itxiak diren arkitekturetan oinarritzen dira, esperimentazio zehatz baten betekizunei erantzuten dutenak. Gertaera honek urrutiko laborategien hedatzea zailtzen du, teknologia honen aireratzea oztopatuz.

Tesi honetan aurkeztutako ikerlanak arkitektura ireki bat proposatzen du. Honekin, landatutako teknologiekin interneten bidez esperimentazioak egitea ahalbidetzen duten urrutiko laborategien zabaltzea erraztuko da, teknologien iraunkortasuna bermatuz.

Arkitektura hau ikerketa-komunitatean hedatuta dauden urrutiko laborategi garrantzitsuenen azterketaren emaitza da. Arkitektura honen baliozkotzerako, nahiko inplementazio burutu dira esperimentazioaren hornidura eraginkorra izateko ezarritako baldintzen betetzea ebaluatzeko. Inplementazio hauetan sartutako urrutiko laborategiak ez daude oinarrituta bakarrik proposatutako arkitektura balioztatzeko kontzeptu frogetan. Honetaz aparte, bere balioa egiaztatu dute unibertsitate-komunitatean erreminta didaktiko bezala.

Agradecimientos

Quiero empezar estos agradecimientos dando las gracias a todas las personas que forman y han formado el equipo de WebLab-Deusto. Han pasado ya muchos años desde que me explicasteis por primera vez los entresijos de la experimentación remota y ha sido un verdadero placer compartir con vosotros los muchos hitos conseguidos. En especial agradecer a Javier García Zubía su acompañamiento continuo durante estos años siendo siempre el mejor ejemplo de compañerismo.

No me quiero olvidar del resto de compañeros de la Universidad, tanto en educación como en investigación, que día a día me recordáis por qué soy un afortunado al dedicarme a esto. Especialmente agradeceros vuestra comprensión estos últimos meses en los que me habéis permitido volcar todos mis esfuerzos en terminar esta tesis, liberándome de otras de mis responsabilidades.

Por supuesto quiero brindar un especial agradecimiento a mis padres por acompañarme siempre y darme su apoyo incondicional. También quiero agradecerles a mis hijos por alegrarme cada día.

Pero sobre todo quiero dedicar esta tesis a Elisa por su amor, confianza apoyo y paciencia. Gracias por darme lo que más quiero y por aguantarme, que no es fácil.

Índice

Índice de Figuras

Índice de Tablas

Capítulo

1

Introducción y contextualización

Los sistemas embebidos son sistemas informáticos que se dedican exclusivamente al control de un sistema mayor sobre el que se integran. La relevancia de los sistemas embebidos crece continuamente. Los computadores se están alejando de los despachos y han pasado a controlar todo tipo de dispositivos de uso cotidiano como tarjetas de crédito, teléfonos móviles, coches y aviones. Su presencia es cada vez más importante en todo tipo de instalaciones como casas, oficinas y fábricas.

Actualmente los sistemas embebidos representan el 98% de los dispositivos basados en microprocesadores que se venden en el mundo. El mercado de sistemas embebidos ha mantenido una tasa de crecimiento anual compuesto (CAGR) superior al 6% durante los años de crisis. En el año 2011 el mercado los sistemas embebidos se valoró en 121.000 millones de dólares (USD) y el estudio de mercado realizado en 2013 por la consultora Transparency Market Research estima que para el año 2018 la cifra llegará a los 200.000 millones de dólares (USD) (Transparency Market Research, 2013). Ese mismo estudio estima que en el año 2020 cohabitarán en el mundo 40.000 millones de sistemas embebidos. Actualmente, cada hogar del primer mundo cuenta con una media de 150 sistemas embebidos, cifra similar a los sistemas integrados en un coche de gama alta (UBM Tech., 2014).

Los sistemas embebidos confieren valor añadido a los productos sobre los que se integran, bien ampliando la gama de funcionalidades ofrecidas o mediante la mejora de la calidad de las funcionalidades tradicionales que se ofrecen al usuario. Las empresas europeas son líderes en el diseño de sistemas embebidos a nivel mundial y tienen una fuerte demanda de especialistas altamente cualificados.

Por otro lado, la continua evolución de las tecnologías utilizadas para el desarrollo de sistemas embebidos, con apariciones constantes de nuevos dispositivos que ofrecen mejor rendimiento y capacidad de interconexión, exige una actualización continua en los contenidos, herramientas y sistemas de desarrollo utilizados para su aprendizaje.

El desarrollo de sistemas embebidos exige un codiseño hardware/software que complica la enseñanza de esta tecnología y su asimilación por los estudiantes (Wolf & Madsen, 2000). En este sentido la experimentación con múltiples tecnologías y dispositivos se vuelve fundamental en la formación de ingenieros con esta competencia (Kumar, Fernando, & Panicker, 2013).

Las principales dinámicas de la educación en ciencia y tecnología se basan en la experimentación en un laboratorio, donde se confirman los conceptos y modelos teóricos y se hacen útiles (Feisel & Rosa, 2005). Existen dos tipos de laboratorios: tradicionales o presenciales, y no tradicionales que incluyen laboratorios remotos y virtuales. Los laboratorios remotos permiten el control de dispositivos reales a través de Internet.

Aunque las primeras experiencias educativas con experimentación remota consistieron en pruebas de concepto, permitieron mostrar el potencial educativo e investigador de los laboratorios remotos, de manera que se propició la incorporación de nuevos agentes e investigadores a la experimentación remota. Se incorporaron investigadores y desarrolladores de países que no solo tenían la tecnología adecuada, sino que además tenían la necesidad real de implementar laboratorios remotos. Así a partir del año 2010 aparecen desarrollos importantes en Australia, LabShare, en Brasil, RexLab o en China, iEELab. Estos desarrollos suponen un nuevo avance tecnológico, pero sobre todo suponen una explotación intensiva de los laboratorios desarrollados ya que en todos los casos se trata de países con extensión geográfica y amplia dispersión demográfica, lo que les convierte en candidatos a usar los laboratorios remotos como forma de acceder a un recurso educativo básico en STEM sin asumir los costes que ello supone en cada centro educativo. Estas iniciativas permitieron que los laboratorios remotos demostraran su interés educativo.

En 2013, la revista Nature (Waldrop, 2013) destacó la oportunidad de utilizar los laboratorios virtuales y remotos en el proceso de enseñanza y aprendizaje. En la

misma línea, también en año 2013, la revista Science (de Jong, Linn, & Zacharia, 2013) publicó un análisis del efecto educativo de los laboratorios virtuales y remotos, concluyendo que "tanto la experimentación virtual como la remota pueden cumplir con los objetivos experimentales en los cursos de ciencias". Esta conclusión está en consonancia con los resultados presentados en (Nickerson, Corter, Eschea, & Chassapis, 2007) y (Corter, Esche, Chassapis, Ma, & Nickerson, 2011) donde se afirma que "los resultados de los laboratorios remotos son comparables en eficacia a los laboratorios presenciales con respecto al objetivo educativo".

Existen multitud de razones que nos llevan a utilizar laboratorios remotos para la enseñanza de los sistemas embebidos. Desde las ventajas económicas y organizativas que surgen de la óptima compartición de los recursos hardware, permitiendo la experimentación con nuevas tecnologías sin necesidad de instalar costosos laboratorios presenciales (García-Zubía & Alves, 2011), hasta las oportunidades que surgen en la compartición de recursos entre distintas instalaciones educativas (Orduña P. , 2013). Por otro lado, hay que destacar su impacto crítico en el aprendizaje a distancia. Muchas universidades ofrecen grados de ingeniería y científicos a distancia, es decir, los estudiantes están siendo certificados en disciplinas donde el trabajo de laboratorio es fundamental para el proceso de aprendizaje.

Resumiendo, en los laboratorios remotos convergen múltiples vectores tecnológicos, didácticos, organizativos y político-administrativos, lo que en cierta medida exige una clara contextualización del trabajo de investigación doctoral.

1.1 Contextualización del trabajo de investigación

Los laboratorios remotos son parte del área de investigación denominada TEL: Technology Enhanced Learning, cuyo objetivo, como indica su nombre, es mejorar el aprendizaje desde la tecnología, o dicho de otra manera, usar la tecnología como herramienta de mejora o intensificación del aprendizaje. El área TEL es el contexto general de esta tesis doctoral, aunque siempre bajo un enfoque tecnológico orientado al desarrollo de sistemas embebidos.

Antes de centrar el contexto tecnológico de la tesis doctoral, es necesario abordar la cuestión del aprendizaje en laboratorios remotos bajo un enfoque pedagógico. Las preguntas centrales son ¿aporta la experimentación remota aprendizaje? ¿es didácticamente comparable al uso de un laboratorio clásico o presencial? ¿es evaluable en términos de aprendizaje? Estas cuestiones han sido objeto de análisis y estudio multidisciplinar por diferentes investigadores, y la conclusión general es que los laboratorios remotos favorecen el aprendizaje con resultados mejores o comparables a los obtenidos mediante un laboratorio clásico.

Esta no es una cuestión central en esta tesis, pero sí que condiciona todo el trabajo, ya que no tiene sentido profundizar tecnológicamente en una herramienta TEL cuya utilidad no ha sido demostrada y aceptada por la comunidad de usuarios e investigadores.

Desde el punto de vista educativo, los laboratorios remotos son vistos como una herramienta de primer orden. Así en el informe quinquenal "Technology Outlook Stem+ 2013-2018" (Johnson, Adams Becker, Estrada, & Martín, 2013) se indica que los laboratorios remotos están encuadrados en la categoría "Time-to-Adoption Horizon: one Year or Less", junto con learning analytics y mobile learning, lo que da una idea de la importancia que los expertos educativos asignan a los laboratorios remotos. Así mismo, en el número 100 del Proceedings of the IEEE (Froyd, Wankat, & Smith, 2012), en el que se analizan 100 años de educación en la ingeniería, se incluyen los laboratorios remotos como uno de los logros del siglo XX.

El contexto específico de la tesis doctoral se centra en los laboratorios remotos para la experimentación con sistemas embebidos publicados en el laboratorio WebLab-DEUSTO de la Universidad de Deusto, ya que sobre ellos se harán las primeras pruebas de idoneidad de la arquitectura planteada como mejora del estado del arte en laboratorios remotos. Pero además, y para sustentar de forma específica su idoneidad, dicha arquitectura también se desplegará en otras plataformas y en otras universidades.

De una forma más general la tesis también tiene una importancia capital en el contexto de la formación a distancia u on-line y más específicamente en los cursos en línea masivos y abiertos (MOOCs) y otros servicios educativos similares. En la actualidad buena parte de las universidades, sobre todo de EE.UU., ofrecen este tipo de formación en sistemas embebidos, pero pocas veces abundan el acceso a experimentos si no es bajo un formato virtual. Pero si se tiene en cuenta que según ABET el uso de laboratorios es un factor crítico en STEM, entonces resulta que el uso de laboratorios remotos es una salida posible a este problema. Esta situación no es preocupante en el caso de cursos no oficiales o no tradicionales (nonformal learning), pero se convierte en un problema para los grados oficiales impartidos on-line. Por ejemplo, en EE.UU. en el año 2014 el porcentaje de estudiantes de educación superior que está cursando al menos una asignatura a distancia es del 33,5% (Allen & Seamann, 2014) lo que ha llevado a las autoridades (policy makers) a preguntarse por cómo se aborda la formación en el laboratorio. En este contexto las universidades y otras instituciones están abocadas a un uso intensivo de laboratorios remotos.

Por otro lado, las altas perspectivas en crecimiento de empleo para ingenieros con competencias en el desarrollo de sistemas embebidos, situadas en un 22% para el año 2018 según un estudio realizado por la oficina de estadísticas laborales en

E.E.U.U. (Lacey & Wright, 2009), han promovido que muchas universidades hayan aumentado el número de materias relacionadas con el desarrollo de sistemas embebidos en los estudios de grado y postgrado. Paralelamente, tanto la industria como los fabricantes de dispositivos para el desarrollo de sistemas embebidos están comprobando la complejidad de encontrar personal con formación de alto nivel en las nuevas tecnologías embebidas. Muchas iniciativas han sido puestas en marcha por fabricantes destinadas a la actualización de profesores y equipamiento en las universidades. Programas universitarios como el Microchip Technology´s Academic Program, el ARM University Program o el Xilinx University Program proporcionan materiales educativos actualizados y descuentos en la adquisición de sistemas de desarrollo software/hardware. Sin embargo la mayoría de las instituciones educativas carecen de recursos para seguir el ritmo de lanzamiento de nuevas tecnologías impuesto por los fabricantes. Justo ahora que acaban de cumplirse 50 años del enunciado de la Ley de Moore, que estima que bienalmente se duplica el número de transistores integrables en un circuito integrado, la aparición de nuevas tecnologías para el desarrollo de sistemas embebidos (multi-core systems, all programable systems on a chip, etc.) requieren una actualización en la metodología didáctica, y esta exige el uso intensivo de los laboratorios. Los laboratorios remotos se postulan como la mejor alternativa para posibilitar a los estudiantes la experimentación con dispositivos para el desarrollo de sistemas embebidos de última tecnología. Para ello es fundamental proporcionar los recursos necesarios para el correcto desarrollo de laboratorios remotos sostenibles que fácilmente sean adecuados a nuevas tecnologías.

En el ámbito tecnológico la tesis se fundamenta en identificar, estructurar, definir funcionalmente y desarrollar los componentes que conforman una arquitectura diseñada para optimizar la publicación de laboratorios remotos dedicados a ofrecer experimentación sobre tecnologías para el desarrollo de sistemas embebidos. Cobrará especial importancia la utilización de dispositivos hardware ligeros que permiten el despliegue masivo y económico de experimentos remotos. Dichos dispositivos ofrecen servicios complejos y potentes a un coste inferior a los 50 euros, lo que contrasta con otros proyectos cuyos costes económicos para cada experimento remoto eran de varios centenares de euros, o de millares.

Resumiendo, en un contexto general TEL la tesis aborda desde una perspectiva tecnológica la forma de permitir el despliegue masivo de experimentos remotos sobre tecnologías embebidas utilizando para ello dispositivos hardware ligeros y económicos.

1.2 Justificación de la investigación

La evolución tecnológica de la experimentación remota ha permitido que cualquier persona o centro educativo tenga acceso a laboratorios remotos para mejorar su proceso de enseñanza/aprendizaje. Es decir, los laboratorios remotos como herramienta TEL popularizan o democratizan un aspecto fundamental de la ciencia: los experimentos.

Esta popularización solo afecta al acceso del experimento remoto, y no a su oferta. Un usuario no solo debe poder acceder a una experiencia remota, sino que también debe ser capaz de proveerla a los demás. De esta forma, la experiencia con laboratorios remotos se enriquece de forma colaborativa. Por ejemplo, un profesor no solo podrá acceder con su aula a un experimento remoto, sino que también podrá remotizar el experimento del que ellos disponen.

Se puede establecer un símil con la gestión de contenidos. Hace años un profesor podía acceder a través de internet a diversas páginas web donde poder descargar contenidos educativos, y de esta forma enriquecer su proceso de enseñanza/aprendizaje. En este escenario la aparición específica de plataformas LMS como Moodle, o la aparición de servicios como Google Drive o Dropbox, permiten al profesor convertirse él también en proveedor de contenidos, creando una comunidad de profesores, y por tanto la separación de roles queda diluida en este proceso que ya es colaborativo y bidireccional. Del mismo modo se espera que un profesor pueda acceder a un experimento ya disponible, o remotizar y compartir el diseñado por él bajo un enfoque de plug&play.

Hasta ahora remotizar un experimento exigía esfuerzo técnico y económico. Por un lado, el diseñador debía tener conocimientos sólidos de comunicaciones, ingeniería del software y tecnología hardware; y además debía disponer de apoyo económico para asumir el coste de los equipos necesarios.

La eliminación de la anterior barrera justifica el desarrollo de esta tesis, y aborda este trabajo desde una doble perspectiva. Por un lado es necesario el desarrollo de una arquitectura que facilite la remotización de experimentos, y por otro lado es necesario el uso intensivo de dispositivos programables de altas prestaciones y bajo coste que permitan el despliegue eficiente de la arquitectura desarrollada.

La tesis afronta el anterior reto desde la perspectiva del usuario final, ya sea el profesor que despliega el laboratorio o el alumno que lo consume, y son los requisitos de este los que caracterizan la funcionalidad requerida de un laboratorio remoto. Es decir, no se busca tanto cumplir ciertos requisitos técnicos sino las necesidades

funcionales del usuario, aunque estas se consigan mediante un uso intensivo de tecnología avanzada en sistemas embebidos. Utilizando como símil la Ingeniería del Software, en ella se utiliza la terminología de requisitos funcionales y no funcionales, aunque en este caso justamente los no funcionales son aquellos que en esta tesis determinan la funcionalidad de los sistemas y por ello se denominan características funcionales. Se indica esto para evitar confusiones en el resto de la tesis, remarcando que en ambos casos se habla de requisitos operativos desde el punto de vista del usuario final.

Es evidente que no es igual remotizar un experimento de química o biología que uno de electrónica o de automatización, y por tanto en esta tesis el trabajo se restringe a los laboratorios remotos que permiten la experimentación sobre tecnologías para el desarrollo de sistemas embebidos. La importancia de la experimentación con estas tecnologías y la dificultad de desarrollar experimentación remota sobre las mismas, así como la necesidad de una adecuación constante a los nuevos dispositivos que emergen en el mercado, justifican esta elección.

Ambos desarrollos -arquitectura y hardware- abrirán una nueva etapa en la implantación de los laboratorios remotos como una herramienta educativa de altas prestaciones.

1.3 Hipótesis y objetivos

Partiendo del contexto ya descrito, la hipótesis de trabajo de esta tesis doctoral se enuncia como sigue:

H.	*Es posible diseñar e implementar a bajo coste una arquitectura que facilite el despliegue eficiente de nuevos laboratorios remotos que permitan la experimentación sobre tecnologías para el desarrollo de sistemas embebidos por cualquier persona, bajo un enfoque que garantice su sostenibilidad.*

La validación de la anterior hipótesis se organiza alrededor de tres objetivos específicos (Figura 1.1):

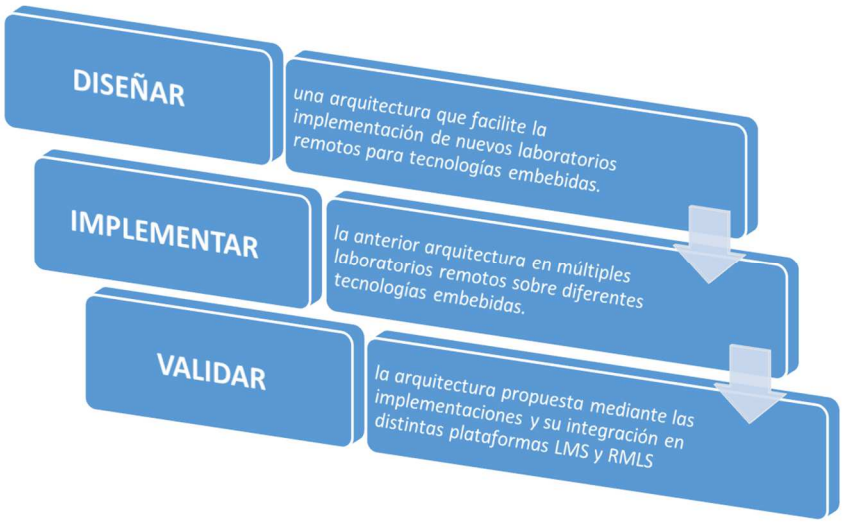

Figura 1.1. Objetivos específicos

OE1. Diseñar una arquitectura que facilite la implementación de nuevos laboratorios remotos (remotización) para experimentación en tecnologías embebidas adaptable a diferentes dispositivos y a múltiples dominios de estudiantes.

OE2. Implementación y despliegue de la anterior arquitectura en laboratorios remotos que puedan funcionar independientemente (standalone), integrados en plataformas software de aprendizaje de tipo LMS (Learning Management Systems) o bajo plataformas de experimentación de tipo RLMS (Remote Labs Management System).

OE3. Validación de la arquitectura propuesta mediante la implementación de múltiples experimentos remotos y su integración en distintas plataformas LMS y RMLS.

Los anteriores objetivos específicos se despliegan en forma de objetivos operativos (Figura 1.2):

OO1. Determinación de los requisitos de la arquitectura que permiten la remotización de experimentos en el escenario de los sistemas embebidos.

OO2. Análisis de las principales implementaciones de laboratorios remotos para la experimentación con tecnologías embebidas en base a los requisitos establecidos en OO1.

OO3. Diseño de la arquitectura según los requisitos establecidos en OO1.

OO4. Identificación de los principales retos tecnológicos que comprometen la implementación de la arquitectura diseñada.

OO5. Implementación de múltiples laboratorios remotos sobre la arquitectura como prueba de superación de los retos tecnológicos establecidos en OO4.

OO6. Selección de los contextos de evaluación de la arquitectura propuesta en base a las implementaciones llevadas a cabo en OO5.

OO7. Evaluación de la arquitectura propuesta en los distintos escenarios elegidos en OO6.

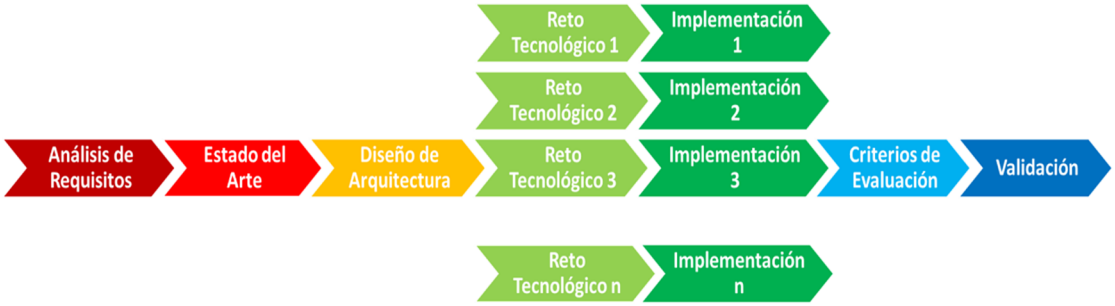

Figura 1.2. Flujo de objetivos operativos.

1.4 Metodología de la investigación

Los anteriores objetivos operativos indicados deben ser alcanzados siguiendo una metodología que asegure su cumplimiento desde un enfoque científico y tecnológico.

El marco principal de la presente tesis corresponde al de investigación aplicada fundamental, siendo el principal objetivo de la misma el diseño de una arquitectura que facilite el despliegue de laboratorios remotos para el desarrollo de experimentación sobre tecnologías embebidas. Sin embargo, la imposibilidad de establecer un modelo cuantitativo que permita la validación de la arquitectura propuesta, será suplida por la implementación de un nutrido conjunto de laboratorios remotos, que permitirán la validación cualitativa de la hipótesis realizada, por lo que la investigación aplicada tecnológica ocupará un papel fundamental en el presente trabajo.

La aplicación de una metodología experimental dentro del método científico, se antoja la más apropiada en función de la naturaleza de la hipótesis planteada. En este sentido dos condicionantes del doctorando han resultado fundamentales para el desarrollo de la investigación llevada a cabo:

1. La participación desde el año 2008 en el equipo de investigación en laboratorios remotos WebLab-Deusto ha permitido contar con la infraestructura, el equipamiento necesario y la accesibilidad a la información y a los expertos en la materia, así como la participación en publicaciones y proyectos de investigación relacionados con el trabajo aquí presentado.
2. La experiencia como docente en cursos relacionados con el desarrollo de sistemas embebidos impartidos en la Facultad de Ingeniería de la Universidad de Deusto ha permitido conocer la metodología didáctica utilizada en laboratorios presenciales y contar con un dominio de estudiantes con el que validar los laboratorios implementados.

La conjunción de ambas experiencias permite al doctorando establecer los requisitos didácticos de los laboratorios remotos y disponer de las herramientas software/hardware para su desarrollo. Este doble perfil de desarrollador y usuario de laboratorios remotos ha resultado fundamental para la ejecución de esta tesis.

El plan de trabajo seguido en el desarrollo de esta tesis se puede observar en la Figura 1.3.

Figura 1.3. Plan de trabajo utilizado en el desarrollo de esta tesis.

Como muestra la Figura 1.4 las fases seguidas durante el desarrollo de esta investigación pueden enmarcarse dentro de una metodología de investigación del tipo investigación-acción, la cual consta de las siguientes fases:

a) Identificación del problema: durante esta fase se determina la problemática y las cuestiones relevantes a solucionar, relacionadas con la problemática a tratar. Se identifican las alternativas posibles, así como las carencias de las soluciones existentes y las posibles mejoras a realizar.
b) Plan de acción: una vez identificado el problema a solucionar, se plantea una posible solución al problema.
c) Desarrollo de acciones: Siguiendo la solución planteada en la fase anterior se implementa un prototipo como solución al problema planteado.
d) Evaluación y observación de los resultados: durante esta fase se observan los resultados obtenidos de las acciones llevadas a cabo y se analizan de manera crítica con el objetivo de evaluarlas y determinar su adecuación a la planificación realizada. Como resultado se pueden tratar nuevas cuestiones que pueden ser relevantes y que ayuden a mejorar el planteamiento inicial.

Figura 1.4. Esquema de la metodología de investigación seguida.

Este método de investigación puede ser repetido de manera cíclica con el objetivo de proporcionar una serie de reflexiones críticas y rigurosas que permitan obtener un proceso de mejora continuo en el que aparezcan nuevos retos e hitos a superar, obteniendo en cada iteración una solución más precisa que la anterior. Es por tanto un proceso de aprendizaje y reflexión cíclico.

1.5 Organización de la memoria

La memoria que describe el trabajo de investigación llevado a cabo de esta tesis se presenta estructurada en un conjunto de capítulos que se corresponde directamente con las fases identificadas en la metodología de investigación.

El capítulo 2, Estado del arte, describe el proceso de experimentación con tecnologías embebidas y establece las características funcionales y de implementación que deben cumplir los laboratorios remotos que ofrecen experimentación con estas tecnologías. El capítulo prosigue con el análisis detallado de las 10 principales implementaciones desplegadas a nivel mundial, incluyendo una comparativa de los 25 laboratorios remotos sobre tecnologías embebidos más relevantes de acuerdo a las características establecidas. Finalmente en este capítulo se identifican las principales arquitecturas utilizadas en el despliegue de los laboratorios analizados. Esta selección será clave a la hora de comparar la arquitectura propuesta con el estado del arte.

El capítulo 3, Especificación de la arquitectura propuesta, incluye un análisis de las arquitecturas existentes y, como parte fundamental de esta tesis, propone una arquitectura específica diseñada en consonancia con las características y requisitos establecidos en el estado del arte. Esta arquitectura dispone los componentes principales que deben desarrollarse en la implementación de un laboratorio remoto para experimentación con sistemas embebidos, definiendo la funcionalidad de cada uno de ellos y la topología a seguir para su interconexión.

En el capítulo 4 se describe un conjunto de implementaciones llevadas a cabo siguiendo la arquitectura diseñada. Cada una de las cuatro implementaciones descritas proporciona una prueba de concepto de la arquitectura propuesta frente a los requisitos establecidos. Dichas implementaciones validarán la idoneidad y bondad de la arquitectura planteada.

Posteriormente a lo largo del capítulo 5 se lleva a cabo la validación de la arquitectura. Metodológicamente la validación consiste en analizar cada una de las implementaciones frente a las tres características fundamentales identificadas en el capítulo 2. El grado de cumplimiento de cada característica se evaluará de forma cualitativa. Finalmente se comparará la arquitectura propuesta frente al estado del arte tomando como referencia las tres características principales: desplegabilidad, escalabilidad y replicabilidad, ya que estas han sido identificadas como las que aseguran la sostenibilidad de los laboratorios remotos con experimentación sobre tecnologías para el desarrollo de sistemas embebidos.

La memoria se completa en el capítulo 6 con las conclusiones obtenidas del análisis y validación del trabajo realizado. En este capítulo se revisan la hipótesis y

objetivos establecidos en el capítulo 1 y se analizan las contribuciones científicas de la solución propuesta junto con las publicaciones realizadas a lo largo del proceso de investigación.

Capítulo

2

Estado del arte

La complejidad de los sistemas embebidos actuales con importantes restricciones de tiempo y consumo, y capaces de gestionar complejos periféricos dificulta enormemente el desarrollo de simuladores y acentúa la importancia de la experimentación para la capacitación en estas tecnologías. Con independencia de la plataforma específica utilizada, el desarrollo de sistemas embebidos requiere el desempeño de un conjunto de fases comunes que perfectamente pueden ser desarrolladas remotamente al no requerir modificaciones en el hardware.

El enorme auge experimentado en los últimos años por los sistemas embebidos y la continua aparición de nuevas plataformas embebidas han motivado la proliferación de laboratorios remotos que permiten experimentar con tecnologías embebidas a través de Internet, permitiendo a estudiantes y desarrolladores experimentar con dispositivos sin disponer presencialmente del equipamiento necesario.

2.1 Introducción

Aunque el desarrollo de sistemas embebidos incluye el diseño e implementación de la plataforma hardware que suele combinar dispositivos electrónicos, eléctricos e incluso mecánicos, es el elemento programable el que consume un mayor tiempo en el ciclo de diseño (Villar, 2001) (Saini, 2012) (Joshi & Gurumurthy, 2014).

Debido al elevado tiempo de desarrollo y al coste de ingeniería no recurrente (NRE) requeridos para la implementación de circuitos integrados de aplicación específica (ASIC), tres son las tecnologías principales que se utilizan para el desarrollo de sistemas embebidos:

- ***Microcontroladores (MCUs)*** son dispositivos de propósito general que permiten el procesamiento de información y control pudiendo ser adaptados a una amplia variedad de aplicaciones. El esfuerzo de desarrollo de aplicación se limita al desarrollo de software y la validación y los costos NRE se amortizan entre todos los usuarios de una arquitectura particular. El rendimiento del sistema está condicionado por la optimización del código, así como el número de transistores requeridos para el almacenamiento. La optimización del código es esencial para lograr la eficiencia de la arquitectura MCU.

- ***Procesadores de señal digital (DSP)*** que disponen de recursos hardware específicos para la implementación de las funciones básicas de muchos algoritmos de procesamiento de señales. Esto optimiza el rendimiento del sistema para las operaciones requeridas, a expensas de la flexibilidad. El código es más simple que el requerido para MCUs. En muchos casos, un DSP es una solución óptima para algunas, pero no todas las funciones requeridas de una aplicación. Muchos MCUs incluyen operaciones DSP básicas en su conjunto de instrucciones, lo que les permite realizar el procesamiento de señal simple, sin la necesidad de un DSP dedicado.

- ***Field Programmable Gate Arrays (FPGAs)*** son dispositivos semiconductores que contienen bloques de lógica cuya interconexión y funcionalidad puede ser configurada mediante un lenguaje de descripción de hardware. La existencia de una gran comunidad de usuarios y la posibilidad de compartir bloques reutilizables limita el esfuerzo requerido para la compleja codificación necesaria para configurarlos. Aportan un elevado rendimiento pero el consumo de energía está lejos de ser óptimo.

A pesar de las importantes diferencias entre estas tecnologías, el proceso de experimentación con las plataformas hardware disponibles para cada una de ellas conlleva el desarrollo de tareas análogas:

1. ***Codificación.*** Este paso exige la edición del programa (MCUs y DSPs) o definición del hardware (FPGAs o PLDs) que gobernará el sistema. Aunque generalmente el resultado de esta tarea resulta en un fichero fuente de texto, existen editores específicos con interfaces asistidas que facilitan el trabajo. Tanto los microcontroladores y procesadores de señal digital, como los dispositivos lógicos programables disponen de herramientas que permiten su programación con diferentes lenguajes. Los MCUs y DSPs suelen ofrecer múltiples lenguajes de programación de alto o bajo nivel (C, ensamblador, BASIC, Java, etc…) y las FPGAs soportan diversos lenguajes de definición de hardware (VHDL, Verilog, etc…)

2. ***Compilación/Síntesis.*** Durante este paso el fichero fuente generado en la tarea anterior se convierte en un fichero binario compatible con la tecnología elegida. Este proceso, compilación o ensamblado en los sistemas basados en microprocesador (MCUs, DSPs, etc…), genera el archivo en lenguaje máquina interpretable por el juego de instrucciones propio de la unidad central de proceso del sistema. En los dispositivos lógicos programables la etapa de síntesis y mapeo

convierte una descripción de circuito en un archivo binario o mapa de celdas (netlist) que contiene la implementación hardware del circuito final.

3. ***Programación.*** En esta etapa el binario generado debe ser programado físicamente sobre el dispositivo final. En el caso de los microcontroladores y procesadores digitales de señal este paso consiste fundamentalmente en ubicar convenientemente el conjunto de instrucciones que forman el programa en la memoria del dispositivo. Para los dispositivos programables la programación se lleva a cabo mediante diferentes tecnologías utilizando células RAM estáticas, transistores EPROM y EEPROM, etc. Este paso suele desarrollarse mediante dispositivos programadores que permiten la programación desde un ordenador mediante un programa a menudo suministrado por el mismo fabricante. El método de programación es uno de los factores determinantes a la hora de desarrollar un laboratorio remoto para la experimentación con tecnologías de sistemas embebidos.

4. ***Validación.*** En esta etapa el usuario debe validar el correcto funcionamiento del sistema desarrollado. La validación de un sistema embebido requiere la monitorización de las salidas y la interactuación con las entradas del mismo. Los sistemas utilizados para gestionar las entradas que nutren al sistema y observar los resultados generados remotamente son así mismo otro componente fundamental de cualquier experimento remoto para el desarrollo de sistemas embebidos.

2.2 Metodología para el análisis del estado del arte

Durante la siguiente sección se van a analizar los principales laboratorios remotos existentes para la experimentación con sistemas embebidos. El análisis se llevará a cabo atendiendo a las *características funcionales*, *aspectos de implementación* y particularidades referentes al *escenario de uso*. Aunque un importante número de laboratorios remotos para experimentación con FPGA, MCU o DSP han sido analizados, a continuación se detallan aquellos más representativos o que proporcionan características de implementación o funcionales que difieren del resto:

- Laboratorio remoto para FPGA de la Universidad de Erlangen-Nuremberg.
- Laboratorio remoto para FPGA de la Universidad de Coimbra.
- "Vicilab" un Laboratorio Remoto para experimentación con circuitos digitales y FPGA desarrollado por la Universidad Nacional de Irlanda en Galway.
- Laboratorio remoto Labshare FPGA Rig soportado por The Labshare Institute.
- CPLD Hybrid Lab de la Universidad de Ciencias Aplicadas de Carintia (CUAS)
- AT89S52 MCU Remote lab de la Universidad de Indonesia.
- Laboratorio de educación a distancia, MicroLab de la Universidad Demirel Sulayman.
- Arduino Remote lab de la Universidad Abierta Helénica (HOU).
- Laboratorio remoto para basado en un DSP de la Universidad de Maribor.

- REAL, Remote Engineering and Applications Laboratory de la Universidad Tecnológica de Illmenau.

Una de los principales requisitos de esta selección de laboratorios remotos para la experimentación con tecnologías de sistemas embebidos ha sido la disponibilidad de los mismos durante el desarrollo de este capítulo. Esto ha permitido seguir una metodología de análisis común para todos ellos en la búsqueda de una comparativa eficiente. Las principales tareas efectuadas en el análisis de cada laboratorio han sido las siguientes:

1. ***Búsqueda y estudio de publicaciones referentes a cada laboratorio***. Se ha llevado a cabo una búsqueda de todas las publicaciones referentes al laboratorio analizado. Está búsqueda ha incluido publicaciones en congresos y revistas de investigación pero también tutoriales y material didáctico asociado.
2. ***Conseguir credenciales para la experimentación***. En algunos casos ha sido suficiente con el registro a través de la web principal del laboratorio, mientras que en otros casos se ha recurrido a los administradores o desarrolladores de los mismos que han colaborado en la redacción de este estado del arte.
3. ***Configuración de un cliente para la experimentación***. Dada la marcada heterogeneidad de los laboratorios analizados esta tarea ha requerido esfuerzos muy dispares. Existen laboratorios que no han requerido acción alguna, mientras que para otros se ha requerido la instalación de entornos de desarrollo para las tecnologías de experimentación, aplicaciones desarrolladas a medida para el laboratorio o la configuración avanzada del navegador desde el que se accede.
4. ***Codificación del diseño***. Utilizado la herramienta de edición recomendada o impuesta por el laboratorio se ha codificado un circuito básico como base para la experimentación. Debido a la falta de documentación relacionada con el uso de algunos laboratorios el desarrollo de un programa compatible con los mismos ha resultado enormemente complejo. Sin embargo, aunque la provisión de documentación sobre el correcto uso del laboratorio es un factor determinante en el éxito de la experimentación, no ha sido reflejado en el siguiente informe al no tener repercusión en la arquitectura de los laboratorios.
5. ***Compilación/Síntesis del experimento***. Durante esta tarea, en aquellos laboratorios en los que se requiere, se ha generado el fichero binario aceptado por el laboratorio mediante las herramientas de desarrollo disponibles para la tecnología de experimentación utilizada en cada caso.
6. ***Programación de la plataforma de experimentación***. A través del laboratorio se ha analizado la herramienta o técnica utilizada para lleva a cabo la programación del dispositivo sobre el que se lleva la experimentación.
7. ***Interacción con el experimento***. Se ha llevado a cabo la validación del experimento utilizando los recursos disponibles del laboratorio.

Para ejecutar estas tareas se han utilizado varios dispositivos que han permitido evaluar la universalidad del laboratorio remoto:

- Portátil Acer Travelmate 8481GT con SO Microsoft Windows 8.1 64 bits
- Portátil Acer Aspire 4810TZG con SO Windows Windows 7 32 bits
- Portátil Dell Latitude 520L con con SO Linux 14.04

- Smartphone LG Nexus 5 con SO Android 5.1
- Smartphone Nokia Lumia 530 con SO Microsoft Windows Phone 8
- Tablet iPAD 2 con SO iOS 8.2

Además durante todas las sesiones de experimentación se ha procedido al almacenamiento y posterior análisis del tráfico HTTP/HTTPS mediante la herramienta Fiddler desarrollada por Telerik. Esto ha permitido calcular los retardos entre las órdenes dadas y las respuestas, así como comprobar los puertos HTTP utilizados y otros requisitos de conectividad requeridos por los laboratorios analizados.

Lamentablemente muchos laboratorios remotos para experimentación sobre sistemas embebidos no han podido ser analizados pormenorizadamente debido a no estar disponibles durante la redacción de esta tesis. Generalmente esto ha sido debido a tareas de mantenimiento, uso exclusivo del laboratorio para el desarrollo de un curso o a la desconexión temporal o permanente del laboratorio. Aun así, al final del capítulo se ha incluido una tabla comparativa de laboratorios remotos para experimentación con tecnologías embebidas, en la que además de los sistemas descritos en detalle se han incluido otros sistemas para los que la biografía disponible permite llevar a cabo su caracterización.

2.3 Caracterización de laboratorios remotos para sistemas embebidos

Para desarrollar un análisis completo de los laboratorios remotos seleccionados, se ha llevado a cabo una caracterización de los mismos en la que se han identificado las principales características que posteriormente permitirán llevar a cabo un análisis completo para desarrollar una comparativa entre los diferentes sistemas. Existen análisis generalistas de laboratorios remotos centrados en la percepción del estudiante desde un punto de vista psicológico (Nickerson, Corter, Eschea, & Chassapis, 2007) (Corter, Nickerson, Esche, & Chassapis, Remote versus hands-on labs: a comparative study, 2004) (Corter, y otros, 2007) (Balamuralithara & Woods, 2009) (Araujo & Cardoso, 2009). Otras caracterizaciones de los laboratorios remotos se enfocan en su capacidad para integración con otras plataformas educativas (Bochicchio & Longo, 2010) (Orduña, 2013) (San Cristóbal, 2010) (San Cristóbal, y otros, 2014). Otros autores han analizado los laboratorios remotos identificando la metodología para la remotización de la instrumentación electrónica (Restivo, Mendes, Lopes, Silva, & Chouzal, 2009) (Hernández-Jayo, 2012). Finalmente existen comparativas que se centran en la forma en la que se conduce la propia experimentación, diferenciando entre la experimentación remota y la presencial (Mougharbel, El Hajj, Artail, & Riman, 2006) (Nedic, Machotka, & Nafalski, 2003). Este análisis se ha llevado a cabo analizando las particularidades de la experimentación con tecnologías para sistemas embebidos.

En esta caracterización se han agrupado las características de los laboratorios remotos analizados en base a tres aspectos generales:

- ***Funcionalidad.***
- ***Implementación.***
- ***Escenario de uso.***

2.3.1 Funcionalidad

La naturaleza de la experimentación con sistemas embebidos torna en fundamentales ciertas características que pueden no resultar trascendentales en otros laboratorios remotos. El breve ciclo de vida de las tecnologías embebidas que son identificadas por el fabricante como descontinuadas u obsoletas tan solo unos pocos años después de salir al mercado exige la actualización periódica de las tecnologías de experimentación. Por otro lado, los tiempos requeridos para la experimentación con sistemas embebidos y la dificultad de los dispositivos de posibilitar la ejecución concurrente de varios experimentos, conlleva la búsqueda de recursos para minimizar los tiempos de espera de los estudiantes. Además el elevado número de dispositivos embebidos está generalizando el uso y compartición de los laboratorios remotos de modo que es fundamental al diseñar un laboratorio remoto facilitar el despliegue del mismo en otras instalaciones para promover la experimentación remota con la misma u otra tecnología.

Atendiendo a la funcionalidad se incluirán todas aquellas características de los laboratorios remotos relacionadas con el consumo de los mismos por parte de los estudiantes. Además, se indicarán todos los aspectos que trascienden a los administradores de los laboratorios para permitir la experimentación.

2.3.1.1 Integración con otras plataformas de aprendizaje o experimentación

Define la forma en la que se gestionan las tareas de administración del laboratorio remoto:

- *Standalone:* El laboratorio funciona directamente con sus propios componentes desarrollados específicamente para soportar el experimento.
- *RLMS:* El laboratorio se conecta a un sistema de gestión de laboratorios remotos que proporciona los principales componentes de integración, autenticación, tracking, publicación de clientes, etc…
- *LMS:* El laboratorio puede ser integrado en una plataforma de gestión de aprendizaje, proporcionando servicios como autenticación, reserva de experimentos, etc.

2.3.1.2 Escalabilidad

Característica que permite la publicación de nuevas instancias de experimentación sobre el mismo laboratorio remoto. En caso de que un laboratorio disponga de esta característica, es necesario indicar si la reserva de cada instancia se realiza independientemente o si el sistema incluye balanceo de carga.

Esta característica es fundamental en la experimentación con sistemas embebidos ya que, debido a los elevados tiempos necesarios para la programación y la validación, no es posible el acceso concurrente de varios usuarios sobre una misma instancia de

experimentación. Por tanto la escalabilidad es la única característica que permite mantener unos tiempos de espera reducidos ante dominios de estudiantes elevados.

2.3.1.3 Replicabilidad

Esta característica indica la posibilidad de exportar el laboratorio remoto hacia otras tecnologías de sistemas embebidos, utilizando la arquitectura implementada. Se identificarán los requisitos de aquellas plataformas de experimentación que permiten la adecuación del laboratorio remoto analizado.

El reducido ciclo de vida de las tecnologías embebidas y el continuo lanzamiento de nuevas familias de dispositivos mejorados por parte de los fabricantes fuerzan la actualización continua de los sistemas de desarrollo utilizados para la experimentación con sistemas embebidos. Por este motivo la replicabilidad se convierte en una característica fundamental en estos laboratorios remotos, siendo necesaria una migración periódica de la plataforma de experimentación para permitir al estudiante la experimentación con los dispositivos demandados por la industria.

Un ejemplo significativo de lo importante que debe ser la replicabilidad en un laboratorio remoto para experimentación con sistemas embebidos se puede observar en el apartado 2.7 de este capítulo. En este apartado se describe el laboratorio para experimentación con FPGA del Instituto Labshare, que en estos momentos se encuentra en plena actualización debido a la discontinuidad, por parte del fabricante (Xilinx), de la arquitectura de la familia de FPGA (Spartan 2) sobre la que hasta ahora se soportaba la experimentación. La complejidad para la adecuación del laboratorio a una nueva tecnología (Spartan 6) ha llevado a retrasar su lanzamiento hasta tal punto que nuevamente el fabricante ha descontinuado la renovada plataforma de experimentación.

Al desarrollar esta caracterización de laboratorios remotos se barajó la opción de incluir como característica la sostenibilidad de los laboratorios. Sin embargo, tras el análisis llevado a cabo en este estado del arte, se concluyó que la sostenibilidad de un laboratorio remoto para experimentación en sistemas embebidos depende directamente de la escalabilidad y de la replicabilidad. La escalabilidad le permite adecuarse a diferentes dominios de estudiantes y la replicabilidad la adecuación a nuevas plataformas de experimentación. Por este motivo, aunque más adelante podamos referirnos a la sostenibilidad de los laboratorios remotos, siempre se hará en relación a estas dos características.

2.3.1.4 Desplegabilidad

Esta característica refleja la posibilidad de desplegar nuevos laboratorios remotos idénticos en otras instalaciones. Hay que identificar los requisitos necesarios tanto de software como de hardware, e indicar la posibilidad de adquirir/fabricar los dispositivos utilizados. Para analizar esta característica es necesario analizar toda la documentación existente que permita el redespliegue del laboratorio.

Esta característica permite a instituciones no involucradas en el desarrollo de laboratorios remotos el despliegue de estos sistemas y la provisión de experimentación remota para sus estudiantes.

En combinación con la replicabilidad, la desplegabilidad permite a cualquier institución educativa el desarrollo de laboratorios remotos para experimentación con las tecnologías embebidas que permitan llevar a cabo remotamente los mismos experimentos que se llevan a cabo en sus laboratorios presenciales.

2.3.1.5 Disponibilidad

Esta característica indica cuándo y cómo el laboratorio es accesible por sus usuarios. Incluye la caracterización del modo de acceso de los estudiantes identificando si el laboratorio remoto dispone de gestión de colas o si es necesario llevar a cabo una reserva previa. Además, se definirán los tiempos máximos de las sesiones, indicando si el sistema permite el acceso múltiple y si la experimentación se lleva a cabo de modo BATCH o se trata de un experimento interactivo. Los laboratorios remotos con experimentación tipo BATCH son aquellos en los que el estudiante solicita la reserva, proporcionando el fichero que contiene el experimento. Cuando llega el turno del estudiante en base a la cola establecida, su experimento es llevado a cabo sin su supervisión y el resultado es enviado posteriormente de vuelta al alumno. Este tipo de experimentos es muy común en aquellos laboratorios en los que el resultado de la experimentación es una gráfica o un determinado valor (del Alamo, y otros, 2002) (Tawfik, y otros, 2013). En los laboratorios remotos para experimentación en sistemas embebidos, este tipo de experimentación es raramente utilizada porque en la mayoría de las ocasiones la validación del experimento requiere de interacción directa con el estudiante.

2.3.1.6 Conectividad

Se analizará la conectividad desde el punto de vista de la arquitectura de comunicaciones en la infraestructura tanto del usuario como del proveedor del laboratorio.

La naturaleza fundamentalmente educativa de los laboratorios remotos provoca que los laboratorios remotos sean comúnmente accedidos desde instituciones educativas. En estos centros, los servicios técnicos informáticos necesitan proteger sus redes informáticas contra virus, malware, el acceso no autorizado y la filtración de datos y para ello suelen recurrir a la instalación de firewalls o sistemas proxy que impiden el acceso a servicios que requieren de configuraciones específicas. Por este motivo es fundamental que el acceso a los laboratorios remotos pueda llevarse a cabo en cualquier instalación sin requerir configuraciones específicas que aumenten la vulnerabilidad de las instalaciones desde las que se publican, así como de aquellas desde las que se acceden.

Para la caracterización de la conectividad se analizarán los permisos especiales que deben proporcionar los servicios IT para posibilitar la publicación del experimento y el consumo del mismo desde otras instalaciones (o la misma). (Puertos, protocolos, direcciones IP estáticas, públicas, etc.).

2.3.1.7 Universalidad

Las nuevas tendencias en la educación online están remarcando la importancia de permitir la experimentación mediante laboratorios remotos desde dispositivos móviles (Orduña P. , García-Zubia, Irurzun, & Rodriguez-Gil, 2011) (Waterson, Landay, & Matthews, 2002) (Terkowsky, Haertel, Bielski, & May, 2013). La experimentación desde muchos laboratorios remotos exige la instalación de determinadas herramientas software que únicamente están disponibles en ciertas plataformas limitando las características de los dispositivos que permiten el acceso a los laboratorios remotos.

En el caso particular de los laboratorios remotos con experimentación en sistemas embebidos, la etapa de compilación o síntesis de los experimentos desarrollados por los estudiantes suele exigir el empleo de un entorno integrado de desarrollo, generalmente proporcionado por el fabricante, que en ocasiones exige unas características de rendimiento y espacio que dificultan su instalación en los dispositivos accesibles por el estudiante.

Algunos laboratorios remotos en sistemas embebidos permiten la ejecución de todas las etapas de la experimentación desde el cliente de los mismos reduciendo los requisitos de los dispositivos desde los que se accede pero aumentando considerablemente los tiempos de cada sesión.

Dentro de esta característica se analizarán, desde el punto de vista del estudiante que accede al laboratorio remoto para llevar a cabo una sesión completa de experimentación, todos los requisitos software/hardware requeridos por el dispositivo desde el cual se lleva a cabo la conexión.

2.3.2 Implementación

En cuanto a la implementación del laboratorio remoto, se analizará la arquitectura del mismo y se identificarán las características relativas a los desarrolladores del sistema así como a los encargados de su despliegue.

Se analizarán los componentes existentes en cada arquitectura y los requisitos para desplegar e interconectar cada uno de ellos.

2.3.2.1 Arquitectura general

La arquitectura general de un laboratorio remoto queda definida por el modo en el que se interconectan los diferentes componentes que lo conforman. Se indicará si los

componentes son independientes o se comunican por protocolos de comunicación que requieren su proximidad y dificultan la desplegabilidad.

2.3.2.2 Servidor

Se analizarán las funciones que ejecuta el conjunto de servidores requeridos para el despliegue del laboratorio. Se indicarán cuáles son las tecnologías y software que utiliza:

- Servidor web.
- SGBD (Sistema de gestión de laboratorios remotos).
- Tecnologías de desarrollo web utilizadas.
- Requisitos a nivel de licencias software (LabView, Citrix, Microsoft Windows, etc.).

2.3.2.3 Servidor de interacción

Una característica particular de los laboratorios remotos para experimentación en sistemas embebidos es la necesidad de interactuar con el propio experimento para la validación del sistema.

Los sistemas embebidos son sistemas electrónicos que analizan el estado de las entradas para desarrollar un correcto control del sistema en el que se integran. Los estudiantes necesitan poder provocar asíncronamente los valores de las señales de entrada y monitorizar el valor de las salidas para testar el funcionamiento del sistema y detectar los posibles fallos.

El servidor de interacción es el sistema que permite la interactuación con el experimento. Para cada laboratorio remoto analizado se indicará su naturaleza (comercial/diseñado a medida), indicando los dispositivos utilizados, el límite de recursos gestionables remotamente (número máximo de GPIOs, DAC, PWM, generador funciones, etc.) y si permite alteraciones de código para su adaptación a otros sistemas (código abierto).

2.3.2.4 Plataforma de experimentación

Se describirá la plataforma sobre la que se experimenta y se indicarán los recursos de esta que están disponibles para los usuarios. Se identificará el método de grabación, las limitaciones de uso y las funciones no replicables remotamente (diferencias con experimentación presencial).

Entre los laboratorios remotos analizados, algunos utilizan los mismos sistemas de desarrollo comerciales que se utilizan en los laboratorios presenciales, mientras que en otros se ha diseñado una plataforma de experimentación a medida enfocada en la experimentación remota. Existen laboratorios que permiten experimentación con

periféricos didácticos (diodos led, displays de 7 segmentos, etc.) y otros posibilitan la experimentación con equipamiento industrial.

2.3.2.5 Cliente

El cliente de un laboratorio remoto incluye todos los componentes que proporcionan al estudiante un interfaz desde el que desarrollar la experimentación. Algunos de los laboratorios remotos analizados utilizan tecnologías web para el desarrollo del cliente, mientras que otros requieren la instalación de un software a medida para acceder al laboratorio.

El análisis del cliente identificará el software necesario para acceder al laboratorio desde cualquier dispositivo. Se indicarán los plugins, applets, programas propietarios y herramientas software de terceros requeridas para llevar a cabo la experimentación.

2.3.3 Escenario de uso

Finalmente se llevará a cabo una descripción pormenorizada del desarrollo de una sesión con cada experimento. Para cada uno de los laboratorios estudiados se indicará paso a paso cuales son las acciones que debe acometer el estudiante para llevar a cabo la experimentación soportada por el laboratorio. Se indicarán los recursos proporcionados para las diferentes etapas identificadas en la experimentación con sistemas embebidos y la forma en la que el estudiante debe acceder al laboratorio.

En este apartado se incluye el análisis de la experiencia de usuario que han podido extraerse de encuestas efectuadas a los estudiantes. Se identificarán además los grupos (dimensiones y características) sobre los que se ejecuta el experimento.

2.3.4 Caracterización de un laboratorio remoto para experimentación con tecnologías embebidas.

La Tabla 2.1 contiene el listado y descripción de las características que serán analizadas durante el análisis de los principales laboratorios remotos destinados a posibilitar la experimentación con sistemas y tecnologías embebidas.

A continuación se analizarán pormenorizadamente de acuerdo a esta caracterización los 10 laboratorios remotos escogidos en base a su relevancia. Además de mostrar las fortalezas y debilidades de cada uno de ellos, este estudio debe permitir establecer cuáles son las características fundamentales de las que depende la eficiencia y eficacia de la experimentación remota en el desarrollo de sistemas embebidos, para priorizar su consecución en la arquitectura que será propuesta.

En este sentido es fundamental analizar los diferentes componentes implementados para el despliegue de cada laboratorio remoto analizado, así como los

mecanismos utilizados para su interconexión. Esto permitirá establecer similitudes entre las diferentes arquitecturas propuestas por la comunidad investigadora, y analizar en cada laboratorio remoto la posible dependencia en la consecución de las características establecidas, en base a las arquitecturas identificadas. Este resultado del análisis de las diferentes arquitecturas utilizadas llevado a cabo en el estado del arte será fundamental en la posterior formulación de la arquitectura que será propuesta en este trabajo doctoral para facilitar el despliegue de laboratorios remotos que soportan la experimentación con tecnologías para el desarrollo de sistemas embebidos.

Tabla 2.1. Resumen de características analizadas en el estado del arte.

	Característica	Descripción
Funcionalidad	Integración	Plataformas software de administración de LR
	Escalabilidad	Número de instancias de experimentación
	Replicabilidad	Adaptabilidad a nuevas tecnologías de SE
	Desplegabilidad	Requisitos para su despliegue en otras instalaciones
	Disponibilidad	Definición del dominio de estudiantes
	Conectividad	Requisitos de la infraestructura de comunicaciones
	Universalidad	Requisitos de los dispositivos clientes
Implementación	Arquitectura	Disposición de los componentes del laboratorio
	Servidores	Servidores de administración y del laboratorio
	Servidor de Interacción	Descripción del sistema que permite interactuar con el dispositivo sobre el que se experimenta
	Plataforma experimentación	Descripción del sistema de desarrollo sobre el que se lleva a cabo la experimentación
	Cliente	Requerimientos software para acceder al LR
Escenario de uso		Resumen de una sesión de experimentación

2.4 Laboratorio remoto para FPGA de la Universidad de Erlangen-Nuremberg

Este laboratorio remoto fue desarrollado por el equipo del Dr. Marc Reichenbach de la Universidad de Erlangen-Nuremberg (Reichenbach, Schmidt, Pfundt, & Fey, 2011). Esta universidad forma parte de la Universidad Virtual de Baviera (Virtuelle Hochschule Bayern, VHB), una red de universidades de ciencias aplicadas del estado libre de Baviera (Alemania) que permite a su alumnado acceder a una variada oferta de cursos online. Desde el año 2011, en el que finalizo su desarrollo, el laboratorio viene siendo usado anualmente en el "Curso básico de FPGA con VHDL" que se imparte desde la plataforma de aprendizaje remota de la Universidad Virtual de Baviera (VHB). Los estudiantes pueden acceder a todos los contenidos didácticos del curso a través del sistema de aprendizaje colaborativo (LMS) de la VHB, basada en la plataforma Moodle. Sin embargo el laboratorio no se encuentra integrado en la misma.

2.4.1 Funcionalidad

El laboratorio permite la experimentación con la plataforma Nexys2 de Digilent (Digilent, 2012) que incorpora una FPGA modelo Spartan 3E fabricada por Xilinx (Xilinx, 2009). A continuación se describen las principales características funcionales.

2.4.1.1 Integración con otras plataformas de aprendizaje o experimentación

Los desarrolladores no han contemplado la integración de este laboratorio con ningún sistema de gestión de aprendizaje (LMS) ni sistema de gestión de laboratorios remotos (RLMS). Los estudiantes pueden acceder a todos los contenidos didácticos del curso a través del sistema de aprendizaje colaborativo (LMS) de la VHB, basado en la plataforma Moodle. Sin embargo el laboratorio no se encuentra integrado en la misma, requiriendo de accesos independientes para ambos sistemas.

Para la gestión del laboratorio remoto, se requiere un servidor ejecuta una aplicación de gestión de recursos desarrollada a medida. Esta aplicación, desarrollada como un "demonio" (daemon) para Linux, se denomina VHBD (Universidad Virtual de Baviera Deamon). Su principal función es arbitrar el acceso de múltiples usuarios a un número limitado de instancias de experimentación hardware. Cada estudiante debe tener la oportunidad de utilizar una instancia del laboratorio y cuando finaliza su experimentación, ésta debe ser liberada. Este sistema impide el bloqueo de una instancia durante largos periodos prohibiendo el acceso de un estudiante durante un corto período de tiempo al finalizar cada experimento.

2.4.1.2 Escalabilidad

El mismo servidor encargado de ejecutar la aplicación de gestión de recursos es capaz de soportar múltiples instancias del experimento remoto. Actualmente el sistema

dispone de 10 instancias conectadas directamente al servidor (Figura 2.1). Cada vez que un estudiante solicita una sesión, el servidor le asigna una instancia libre, o si están todas ocupadas le mantienen en una cola de espera hasta que se libere cualquiera de ellas. La arquitectura soporta un número mayor de instancias, estando éste limitado por la capacidad del servidor de gestionar el streaming de las respectivas webcams conectadas por USB al servidor del laboratorio.

Paralelamente el laboratorio remoto al completo puede replicarse en la misma instalación, si bien la aplicación de gestión de recursos carece de soporte para balanceo de carga entre distintos servidores. En este escenario los usuarios debería escoger el servidor matriz con el que se conectan, pudiendo acceder únicamente a las instancias alojadas en este.

Figura 2.1. Fotografía del laboratorio remoto para la experimentación con FPGAs de la Universidad de Erlangen-Nuremberg.

2.4.1.3 Desplegabilidad

Todas las herramientas software requeridas para el desarrollo del laboratorio son Open Source. Esto posibilita el despliegue del laboratorio en otras instalaciones, si bien los autores no proporcionan información sobre la instalación del sistema.

Así mismo, aunque la plataforma de experimentación es fácilmente adquirible desde su fabricante, y lo mismo ocurre con el dispositivo empleado para la implementación del servidor de interacción, los autores de este laboratorio han desarrollado una tarjeta impresa que permite la conectividad entre ambos dispositivos sin publicar los archivos que permiten su fabricación.

2.4.1.4 Replicabilidad

La elección del software abierto UrTAG para la programación de la plataforma de experimentación facilita enormemente la replicabilidad del laboratorio sobre otras plataformas que permiten su programación mediante JTAG (Joint Test Action Group). Este estándar (IEEE Computer Society, 2013), diseñado para el testeo de tarjetas de circuito impreso, ha sido adoptado por compañías electrónicas de todo el mundo y se emplea para la programación de infinidad de sistemas embebidos (FPGAs, MCUs y DSPs). La independencia del servidor de interacción y de la webcam con la plataforma de experimentación facilita la sustitución y actualización de esta, e incluso la convivencia de instancias basadas en distintas tarjetas de experimentación. El único requisito necesario para ello es que cada instancia disponga de un identificador de producto/vendedor diferente permitiendo su diferenciación mediante el software UrTAG.

Obviamente la tarjeta de interconexión entre el servidor de experimentación y la plataforma de experimentación, diseñada a medida para la Nexys 2, debe ser modificada si desea ser adaptada a otra plataforma.

2.4.1.5 Disponibilidad

El laboratorio está disponible para todos los estudiantes de las universidades que pertenecen la VHB, requiriendo la matriculación gratuita en el "Curso Básico de FPGAs con VHDL".

Los estudiantes pueden acceder en cualquier momento y reservar una instancia libre o esperar a que se libere una de ellas. Este laboratorio carece de sistema de reserva previa. Si un estudiante desea utilizar una instancia, el servidor busca la primera libre en la tabla de instancias y la asigna para el usuario. El usuario obtiene un ID de sesión y tiene una cantidad limitada de tiempo para experimentar. Si el tiempo ha terminado, o el usuario ha cerrado la sesión, la instancia quedará liberada y podrá ser asignada de nuevo a otro usuario que la solicite.

2.4.1.6 Conectividad

Desde el punto de vista de la conectividad este laboratorio presenta ciertos requisitos que requieren la apertura de permisos por parte del servicio informático de la institución. Cada instancia del laboratorio requiere utilizar tres clientes simultáneos:

1. Conexión SSH con el servidor del laboratorio. La reserva del experimento y la programación del dispositivo se lleva a cabo mediante una sesión SSH realizada mediante el puerto 22. Aunque el servidor dispone de la seguridad que ofrece el propio sistema operativo (Linux) y los usuarios disponen de permisos limitados, un servidor SSH supone un peligro para cualquier red, fundamentalmente porque los nombres de usuario y contraseñas viajan en formato plano.

2. Conexión HTTP con el servidor de interacción. El servidor de interacción consiste en un microservidor web que permite alterar el estado de sus salidas (conectadas a la entrada de la plataforma de experimentación) desde una página web. El acceso a la web se lleva a cabo mediante una conexión túnel desde el servidor principal a través de un puerto que oscila en función de la instancia entre el 8001 y el 8010, lo cual puede complicar tanto el acceso como la desplegabilidad en instituciones externas.
3. Sesión Video Lan Client (VLC). El streaming de las webcams se lleva a cabo desde una sesión VLC, que se desarrolla también mediante una conexión túnel con el servidor principal en los puertos 8101 al 8110. El requisito de apertura de estos puertos puede suponer un problema en aquellas instalaciones que se conectan a Internet por detrás de un firewall.

2.4.1.7 Universalidad

Desde el punto de vista de los dispositivos que permiten el acceso al laboratorio, hay ciertas características que limitan las opciones. Sin entrar en aspectos que perjudican la usabilidad, las tres conexiones referidas en el apartado anterior pueden llevarse a cabo con las principales plataformas, pudiendo encontrar clientes SSH, web y VLC compatibles con los principales sistemas operativos para ordenadores personales como Windows, Linux, MacOS, etc… e incluso, para plataformas móviles como Android, iOS, Windows Phone, Ubuntu Phone, etc….

En este caso la principal limitación en cuanto a las características de los dispositivos cliente es sin duda la necesidad de realizar las tareas de síntesis y mapeo del circuito en el propio dispositivo cliente. Esta acción requiere tener instalado la suite Xilinx ISE, que es compatible únicamente con las plataformas Windows y Linux y exige un rendimiento mínimo considerable (4Gb RAM y 5GB HDD).

2.4.2 Implementación

La arquitectura de este laboratorio puede observarse en la Figura 2.2.

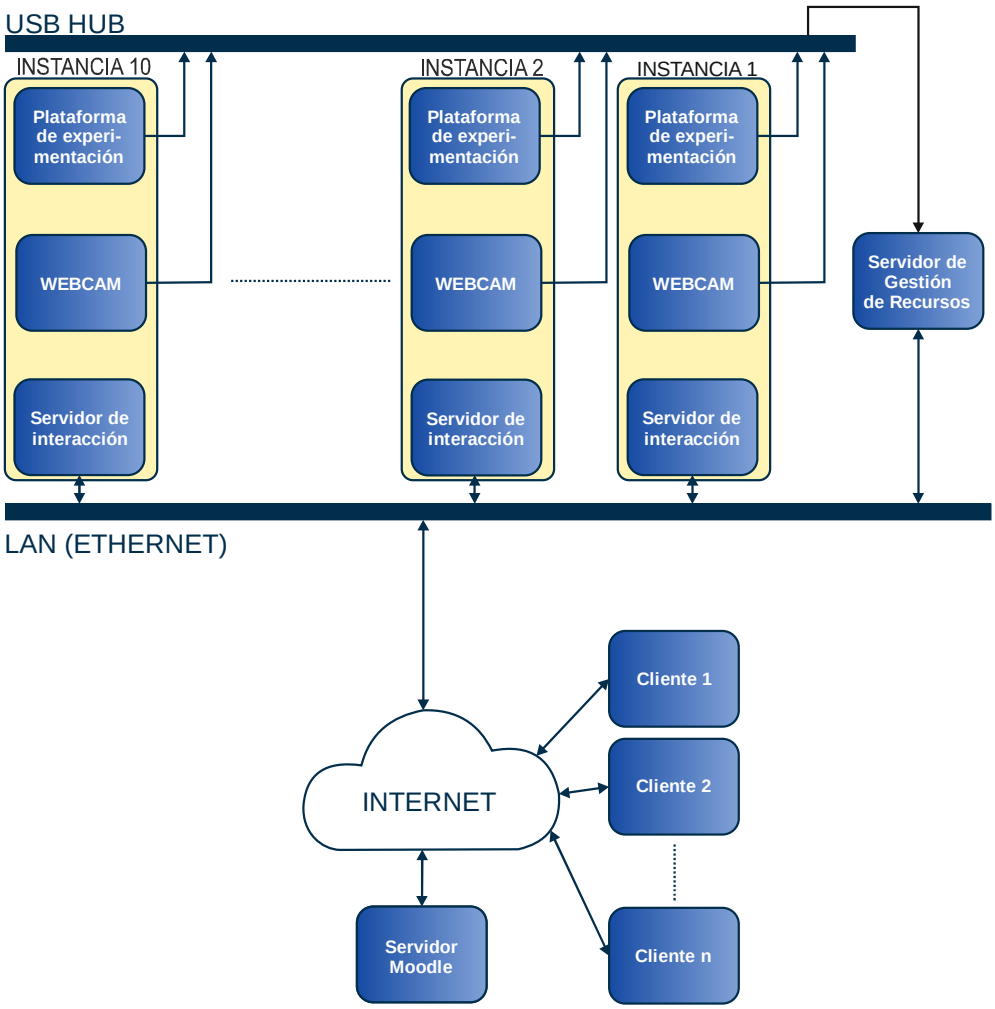

Figura 2.2. Arquitectura del laboratorio remoto para la experimentación con FPGAs de la Universidad de Erlangen-Nuremberg.

2.4.2.1 Servidor de recursos

El elemento principal del laboratorio está formado por el servidor de gestión de recursos. Este servidor está implementado sobre sistema operativo Linux, utilizando la distribución Gentoo Linux para proporcionar la máxima seguridad y flexibilidad. Este servidor se encarga de un importante número de tareas:

1. *Autenticación de usuarios*. Los usuarios del laboratorio remoto reciben unas credenciales que permiten el acceso al servidor utilizando el protocolo SSH con la seguridad propia del sistema operativo.

2. *Asignación de instancias a usuarios*. Un demonio Linux desarrollado a medida del experimento se ejecuta en el servidor. Este demonio, denominado VHBD, se encarga de asignar las instancias del experimento a un único usuario, proporcionando a éste un puerto de acceso al servidor de interacción y a la webcam de la instancia reservada. El VHBD está programado de manera modular. Cada componente del sistema es un proceso propio que se comunica a través de IPC (pipes, colas, etc…). De esta manera, el sistema se puede ampliar fácilmente con más instancias e incluso,

modificando el VHBD para sustituir los pipes por sockets, más servidores. Este sistema se ha desarrollado en cooperación con la Universidad de Passau. Cada usuario tiene su propio proceso, permitiendo que muchos usuarios pueden trabajar en el sistema en paralelo.

3. *Programación de las instancias de FPGA*. Todas las tarjetas Nexys 2 están conectadas al servidor a través de un hub USB. La programación se lleva a cabo mediante el software abierto UrTAG. Le elección de este software implica la reprogramación del chip Cypress FX2 integrado en la tarjeta Nexys2 con un firmware desarrollado por terceros. Además para identificar cada una de las instancias, estas deben ser programadas con un PID (identificador de producto) diferente.

4. *Streaming de webcams*. Para la monitorización del hardware, se envía al usuario un streaming de vídeo. Las webcams están conectados a través de Hub USB al servidor, que también tiene que diferenciar las diferentes cámaras mediante un número de serie único provisto por el fabricante (Logitech). Los dispositivos autogenerados por Linux para cada cámara (/dev/videoX) son asignados mediante udev a dispositivos lógicos (/dev/vhbcamx) y gestionados mediante "video for linux 2" (v4l2). Se ha elegido una codificación MPEG-2 con un bitrate de 400 kbits/seg. Esto exige un ancho de banda requerido para cada instancia, pero proporciona la calidad suficiente.

5. *Generación de estadísticas*. Todas las acciones llevadas a cabo por los usuarios en cada sesión de experimentación deben quedar registradas. Esto permite a administradores y profesores visualizar el número de accesos, la duración de las sesiones y los fallos detectados durante las mismas.

2.4.2.2 Servidor de interacción

El servidor de interacción se ha implementado sobre una tarjeta Celeritous PICWEB Server. Para cumplir los requisitos marcados, el servidor de interacciones debe cumplir tres características:

1. Permitir escribir y leer los valores de los pines conectados con el experimento. Esto ha requerido modificar el servidor genérico proporcionado por la tarjeta, incluyendo una petición HTTP que permite la lectura de los pines. Para establecer los valores de los pines de entrada de la FPGA, el usuario tiene que conectarse a la página web alojada en el servidor de interacción que emula los interruptores y pulsadores mediante una interfaz virtual.

2. Incluir seguridad que impida a usuarios no asignados a una instancia acceder a la web de interacción. Para ello se ha introducido el concepto de los identificadores de sesión. Una secuencia alfanumérica aleatoria de caracteres, ofrecida por el servidor de recursos en la propia URL que ofrece a los usuarios, es exigida para el control de un servidor de Interacción.

3. Generar pulsos de acuerdo con el protocolo PS/2 para simular un teclado. Para ello, se ha implementado un generador de secuencias que produce secuencias de protocolo PS/2 que los usuarios pueden generar a través de la interfaz web.

Para la conexión de la tarjeta de interacción y el experimento, se ha desarrollado una tarjeta de circuito impreso en una tecnología de dos capas que aporta por cada pin un indicador led (Figura 2.1). El servidor de interacción proporciona la emulación de 8 interruptores, 4 pulsadores y 2 hilos para la emulación del protocolo PS/2.

2.4.2.3 Cliente

Para poder acceder al servidor el cliente requiere disponer de tres programas:

1. Navegador web para acceder a la página web servida por el servidor de interacción.
2. Cliente SSH para conectarse con el servidor de gestión recursos.
3. Cliente VLC para visualizar el streaming de la webcam.

Además el cliente debe tener acceso a un conjunto de puertos que pueden vulnerar las políticas de los servicios informáticos de algunas instalaciones. En concreto son los puertos 22, 8001-8010 y 8101 a 8110.

2.4.2.4 Plataforma de experimentación

La experimentación final de los estudiantes se desarrolla sobre la tarjeta comercial Nexys 2, fabricada y comercializada por Digilent Inc. El lanzamiento de esta tarjeta se llevó a cabo en 2009 con fines fundamentalmente didácticos. La Nexys 2 es una potente plataforma de diseño de sistemas digitales basada en una FPGA Xilinx Spartan 3E. Dispone de 16Mbytes de SDRAM y 16Mbytes de flash ROM. La plataforma Nexys 2 es ideal para la implementación de procesadores embebidos soft como el microprocesador de 32-bit RISC Microblaze ™ ofrecido gratuitamente por Xilinx. Además, esta plataforma integra un puerto USB 2.0 y dispone de una amplia colección de dispositivos de E/S, puertos de comunicación y conectores de expansión.

2.4.3 Escenario de uso

Para experimentar con este laboratorio remoto el usuario debe llevar a cabo los siguientes pasos:

1. Generar el fichero de programación para el experimento. Este paso requiere que el usuario disponga de la suite informática Xilinx Webpack ISE. Este software no es abierto pero es suministrado por Xilinx gratuitamente. Incluye un Project IDE que permite llevar a cabo todos los pasos desde la edición de la definición del hardware hasta la generación del fichero de programación. Desde el servidor de contenidos (Moodle) donde se publican los materiales del curso, los estudiantes pueden descargarse un fichero UCF que define los pines de la FPGA donde los periféricos están conectados y debe ser incluido en cada proyecto generado por los estudiantes.
2. Una vez los estudiantes han generado el fichero de programación correspondiente al experimento, el siguiente paso establecer una comunicación SSH con el servidor

de gestión de recursos (faui3vlab.informatik.uni-erlangen.de). Si alguna de las instancias se encuentra libre en ese momento la conexión se establece y permite la programación de la FPGA reservada. Para ello el estudiante debe primero copiar el fichero de programación mediante scp y posteriormente ejecutar un script que lleva a cabo la programación mediante UrTAG.

3. En caso de que la programación de la FPGA se realice con éxito, el cliente SSH proporcionará dos direcciones URL que permiten acceder respectivamente a la web alojada en el servidor de interacción y al streaming de video generado por la webcam.

4. En una ventana del navegador mediante la primera URL el estudiante accede a una web, que mediante un interfaz gráfico permite emular los 8 interruptores, los 4 pulsadores y el teclado PS2.

5. Finalmente, el usuario deberá abrir un cliente de VLC media player con la segunda URL provista por el servidor de gestión de recursos para monitorizar la imagen generada por la webcam de la instancia reservada.

Una vez realizados estos pasos el estudiante puede interactuar con el experimento con un interfaz gráfico de usuario resultante de la combinación de las tres herramientas cliente. En la Figura 2.3 se muestra una disposición posible de las herramientas requeridas para la experimentación en este laboratorio remoto.

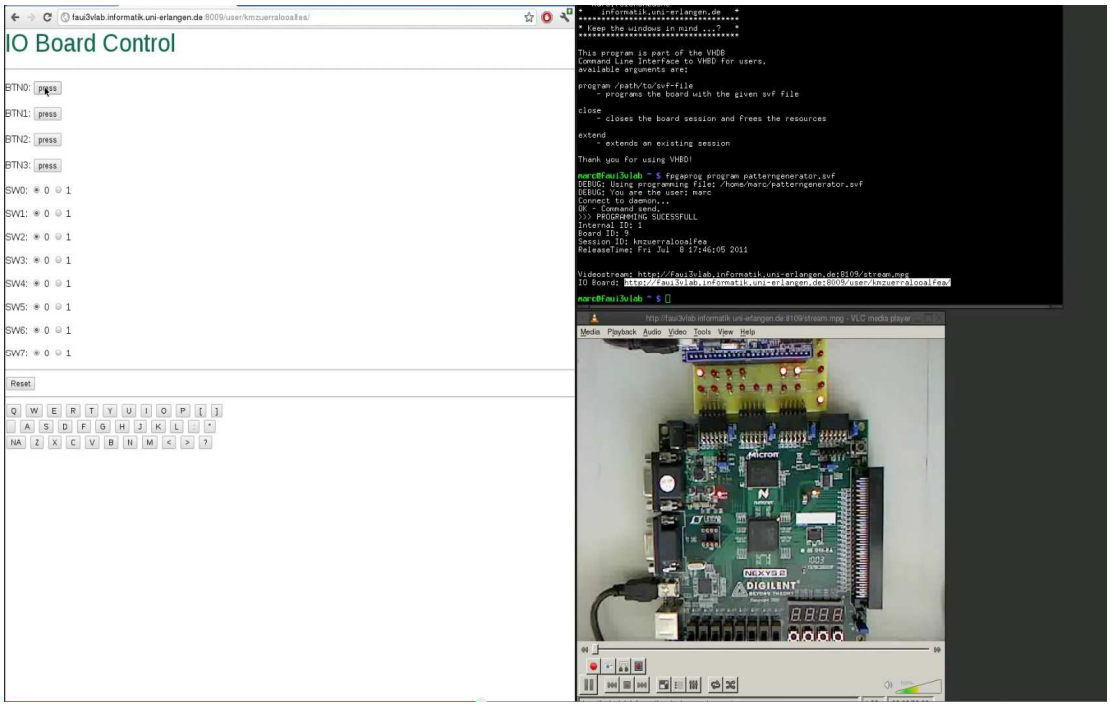

Figura 2.3. Ventana del cliente para el laboratorio remoto para la experimentación con FPGAs de la Universidad de Erlangen-Nuremberg.

2.4.4 Resumen

Este laboratorio remoto permite experimentar con una FPGA Spartan3E de Xilinx, interactuando con las entradas del sistema y monitorizando el desarrollo del circuito mediante la imagen generada por una webcam.

Las grandes ventajas de este laboratorio son la disponibilidad, pues lleva funcionando de manera prácticamente ininterrumpida desde el año 2011 y la escalabilidad, contando actualmente con 10 instancias idénticas que permiten la experimentación simultanea de dicho número de usuarios.

Aunque los principales componentes necesarios para su despliegue son productos comerciales, la compleja configuración de los mismos dificulta su desplegabilidad. Además, aunque el desarrollo del laboratorio se ha llevado a cabo mediante la utilización de software de código abierto, alguna de las aplicaciones software desarrolladas para la gestión del laboratorio no han sido publicadas. El requisito de apertura de 20 puertos HTTP libres (2 por instancia), dificulta su despliegue en instalaciones donde directivas de seguridad rigen la labor de los servicios informáticos.

Paralelamente desde el punto de vista de usabilidad los usuarios deben tener conocimientos del intérprete de comandos Linux, siendo necesario establecer una sesión SSH y ejecutar una serie de scripts sobre el propio servidor principal del laboratorio. Además para interactuar con el laboratorio se requieren tres clientes simultáneos que empeoran la experiencia de usuario y limitan su acceso desde plataformas móviles.

En cuanto al coste del laboratorio, el requisito marcado por los desarrolladores de utilizar software abierto elimina todo el posible coste generado por licencias. En cuanto al hardware, tanto la plataforma de experimentación (Nexys 2, 270€) como el servidor de interactuación (Celeritous PICWEB, 68€) y la webcam (Logitech C500, 25€) han sido escogidos para abaratar el coste de cada instancia del experimento, que incluyendo la tarjeta de interconexión entre la plataforma de experimentación y el servidor de interactuación, implementada a medida para el laboratorio, no supera un total de 400€. El alto rendimiento requerido para el servidor de gestión de recursos (1000€) y el enorme número de puertos USB requeridos que implica el uso de un concentrador de puertos (400€) encarece ligeramente el coste de este laboratorio. Aun así, el coste total del laboratorio, que ronda los 6000€, puede ser considerado como muy comedido teniendo en cuenta que permite la experimentación simultanea de 10 usuarios gracias a las 10 instancias replicadas.

2.5 Laboratorio remoto para FPGA de la Universidad de Coimbra

Este laboratorio remoto ha sido desarrollado por el "Instituto de Sistemas y Robótica" de la Universidad de Coimbra, bajo la supervisión del Dr. Jorge Lobo (Lobo, 2011) (Soares & Lobo, 2011) (Gomes, Patricio, Ferreira, & Costa, 2009). La intención de los

desarrolladores ha sido permitir a los estudiantes desarrollar experimentación remota en diseño digital, permitiendo probar circuitos digitales bajo una plataforma FPGA. El objetivo de este proyecto es hacer un uso más eficiente de los recursos del laboratorio a través de Internet. Los estudiantes pueden probar el funcionamiento real de circuitos digitales como alternativa a la simulación. Los estudiantes pueden interactuar con el experimento remotamente, monitorizando las salidas de los circuitos a través de una webcam y controlando las entradas mediante botones virtualizados en un sencillo panel de control de conmutación. El laboratorio incluye una plataforma que permite el almacenamiento de proyectos y diseños que pueden ser ejecutados mediante el laboratorio remoto.

2.5.1 Funcionalidad

El laboratorio DE2 web Portal permite interactuar con el sistema de desarrollo DE2 de Altera (Altera, 2012), permitiendo la experimentación con la FPGA Altera Cyclone II (Altera, 2008).

2.5.1.1 Integración con otras plataformas de aprendizaje o experimentación

El laboratorio remoto DE2 web Portal es completamente independiente, no estando contemplada su integración con sistemas de gestión de aprendizaje (LMS) ni sistemas de gestión de laboratorio remotos (RLMS).

El portal web desarrollado para gestionar el laboratorio incorpora su propio sistema de autenticación de usuarios y herramientas de administración, que permiten al administrador añadir y borrar usuarios y acceder a estadísticas de acceso por parte de los usuarios.

El portal del laboratorio incorpora una librería de proyectos accesible por el usuario donde puede descargarse implementaciones para su testeo mediante el laboratorio remoto. Asimismo, existe una página de concursos que ofrece al usuario cuestionarios y animaciones en FLASH para reforzar sus conocimientos en diseño de circuitos digitales.

2.5.1.2 Escalabilidad

Este laboratorio no contempla la inclusión de nuevas instancias, y ningún sistema de balanceo de carga entre diferentes servidores ha sido planteado.

Tampoco existe la posibilidad de incorporar varias instancias en un mismo servidor. La comunicación con la plataforma de experimentación se lleva a cabo directamente mediante un puerto USB del servidor a través de la línea de consola de la suite Quartus II para la programación de dispositivos programables fabricados por Altera. Esta circunstancia impide la conexión simultánea de más de una tarjeta DE2 ya que el software carece de capacidad para identificar cada una de ellas.

2.5.1.3 Desplegabilidad

Aunque la programación de la plataforma de experimentación se lleva a cabo mediante una aplicación de software propietario (Quartus II), Altera permite su descarga y utilización gratuito sin limitaciones. El resto componentes software requeridos por el servidor de gestión del sistema se componen de un servidor web apache2 con módulo PHP y un servidor desarrollado en Java que permite remotizar el streaming de la webcam, desarrollado por Yawcam y puede ser descargado libremente desde Internet. Lamentablemente tanto el software Altera Quartus II, como el servidor de streaming para la webcam (Yawcam) están disponibles únicamente para plataformas Windows, limitando las opciones del sistema operativo del servidor principal del laboratorio.

En cuanto al hardware, la principal característica de este laboratorio remoto es que utiliza la propia plataforma de experimentación para implementar el recurso que permite al estudiante interactuar con el experimento, alterando las entradas del sistema dinámicamente. Esta característica reduce el número de dispositivos requeridos para desplegar el laboratorio remoto a un servidor, la plataforma de experimentación (Altera DE2) y la webcam (Figura 2.4).

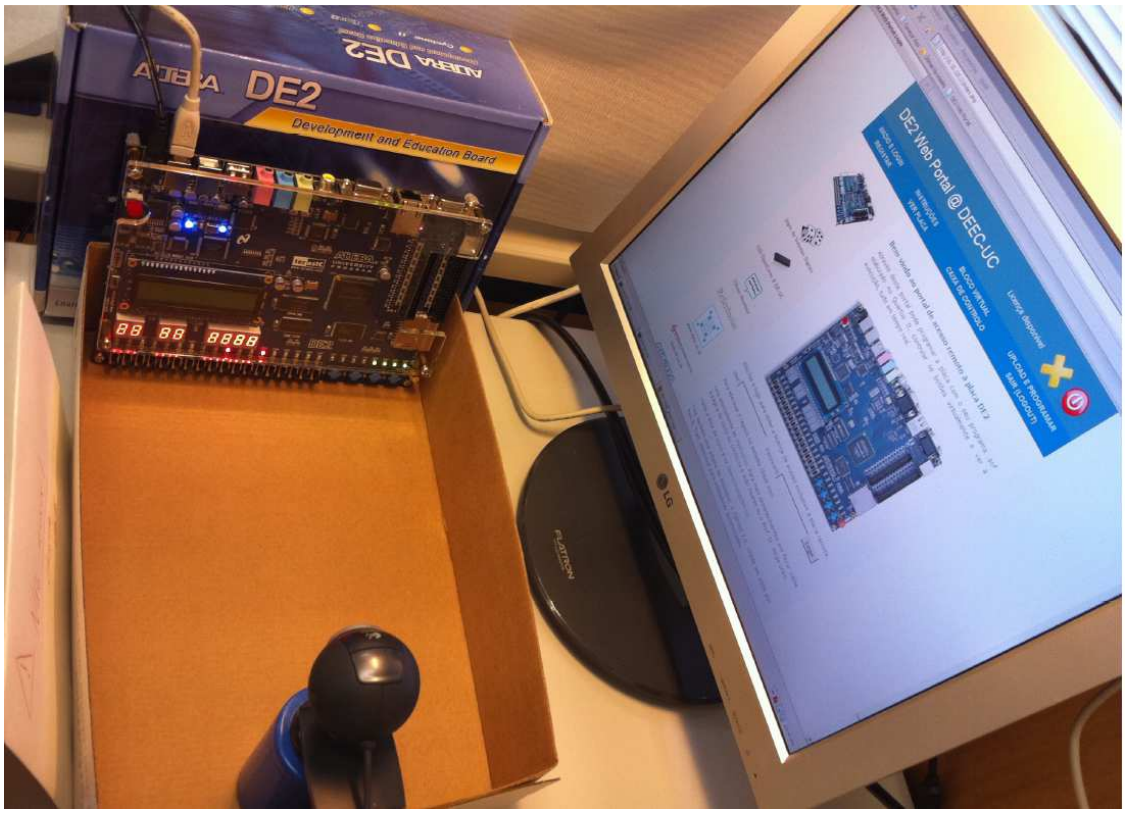

Figura 2.4. Equipamiento necesario para el despliegue del laboratorio remoto DE2 web Portal de la Universidad de Coimbra.

2.5.1.4 Disponibilidad

Este laboratorio remoto ha sido desarrollado para permitir a los estudiantes de la Universidad de Coimbra desarrollar experimentación real en la asignatura de Diseño de

Circuitos Digitales. Cualquier alumno de la Universidad de Coimbra puede solicitar su acceso para el laboratorio remoto. El sistema de registro permite la solicitud de credenciales por estudiantes ajenos a la institución, condicionando siempre su acceso a la disponibilidad del laboratorio.

El servidor de gestión del sistema se encarga de conceder el control del laboratorio a un estudiante. Cada usuario dispone de 5 minutos para llevar a cabo su experimentación. Transcurrido ese tiempo la sesión finaliza automáticamente y el laboratorio se libera. El servidor de gestión del sistema implementa una cola que regula el acceso secuencial de los usuarios en función del orden en el que han accedido.

2.5.1.5 Conectividad

Aunque las principales tareas de acceso y programación son llevadas a cabo desde el portal web del laboratorio, una vez la reserva del mismo es asignada a un usuario existen ciertos requisitos que condicionan la interacción con el experimento.

Esta interacción se lleva a cabo monitorizando el estado de la plataforma de experimentación (tarjeta DE2 de Altera) a través del streaming de una webcam y gestionando el valor de las entradas que alimentan la FPGA desde un panel de interruptores y pulsadores virtualizados. En el caso de la webcam el streaming se lleva a cabo mediante un applet de Java que requiere el plugin apropiado, la instalación del entorno de ejecución de Java (JRE) y la reducción de la seguridad en los navegadores para permitir su ejecución, debido a que el applet no se encuentra firmado (Figura 2.5). Además la independencia de ambos clientes impide la interacción desde un único interfaz gráfico integrado (Reimers & Stewart, 2015). Por otro lado el laboratorio requiere conectividad por un puerto específico para acceder a la webcam que en muchas instituciones pueden ir en contra de las políticas de seguridad del servicio informático.

2.5.1.6 Universalidad

Las principales limitaciones que podemos encontrarnos desde el punto de vista de los clientes con los que accedemos al experimento se refieren al método en el que se desarrolla la monitorización del experimento. Esta se lleva a cabo mediante un applet no firmado, siendo incompatible con algunos navegadores o requiriendo la configuración de las políticas de seguridad de aquellos con los que sí es compatible.

En cuanto a la "Caja de Control" que permite modificar el valor de las entradas del sistema, requiere su apertura en una ventana del navegador independiente complicando la interacción desde dispositivos móviles.

Además dado que el laboratorio remoto recibe los circuitos a implementar en formato SRAM Object Files (.sof) es necesario la instalación del programa Quartus II en el sistema cliente, siendo esta aplicación únicamente compatible con plataformas Windows.

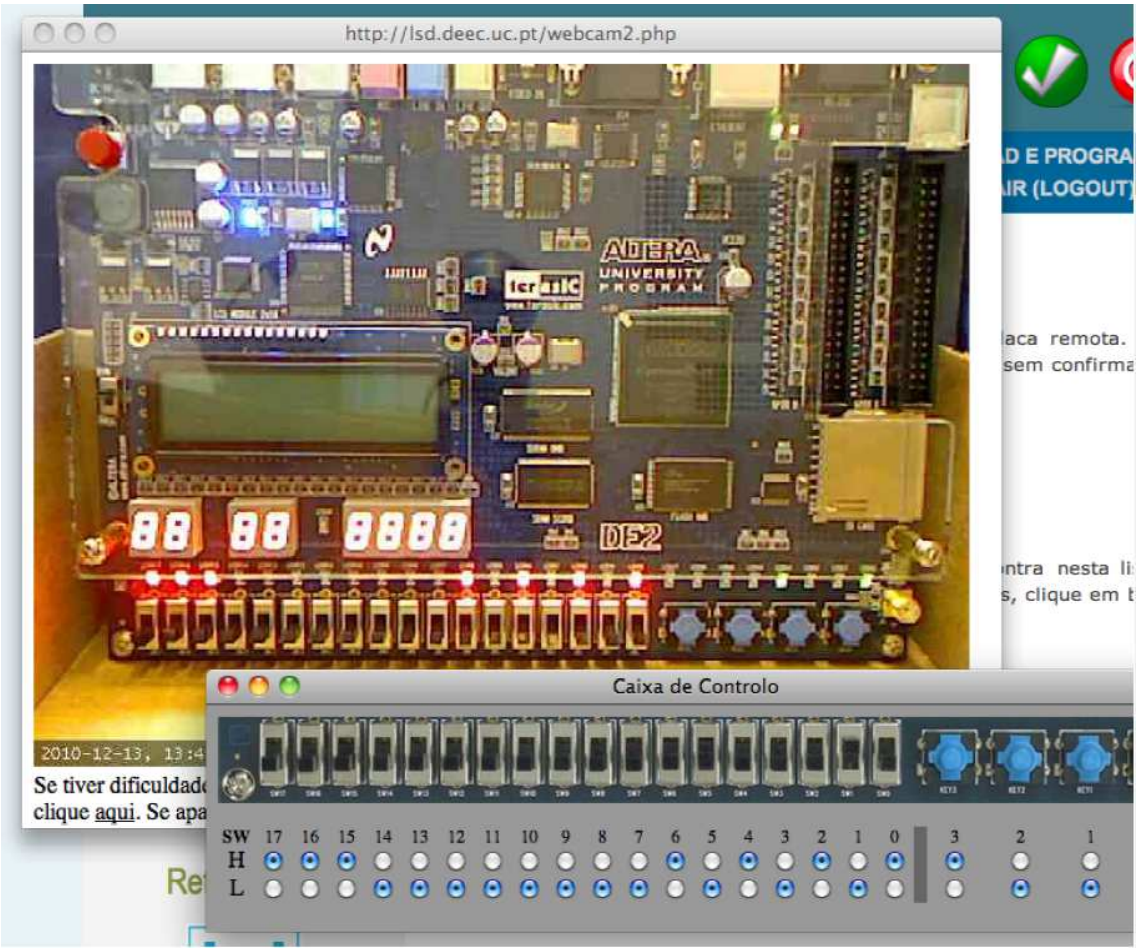

Figura 2.5. Captura de los componentes necesarios para la interacción con el laboratorio remoto DE2 web Portal, desarrollado por la Universidad de Coimbra.

2.5.2 Implementación

La Figura 2.6 muestra la arquitectura de este laboratorio remoto. Un servidor principal de gestión se encarga de todas las tareas relacionadas con la administración, programación, monitorización e interacción con el laboratorio, convirtiéndose en el núcleo principal del laboratorio remoto.

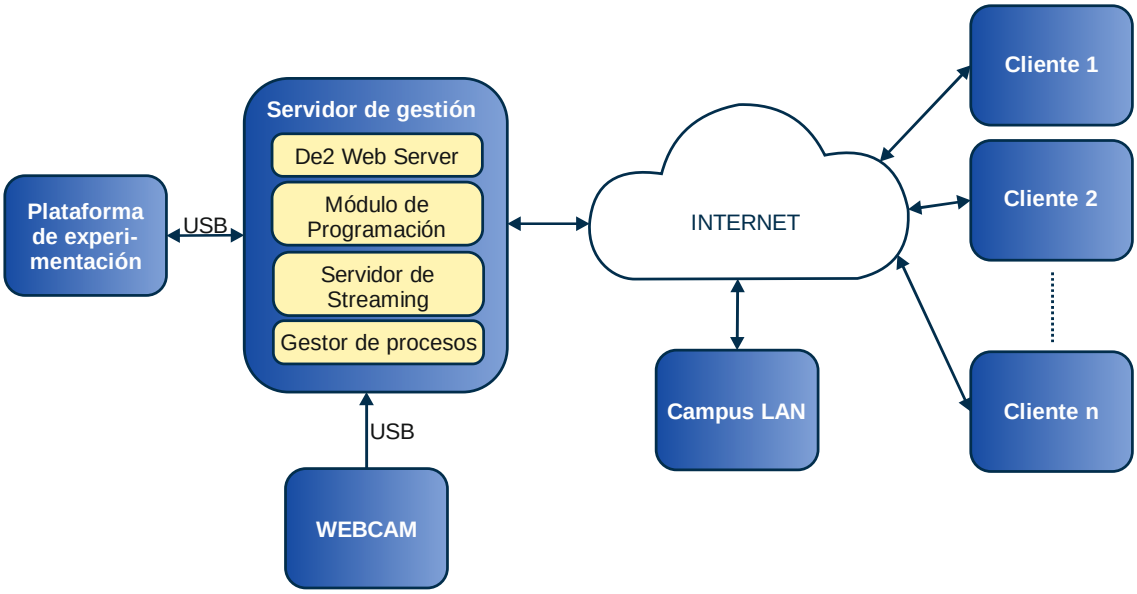

Figura 2.6. Arquitectura del laboratorio remoto DE2 web Portal, desarrollado por la Universidad de Coimbra.

2.5.2.1 Servidor de gestión

En este sistema reside la gestión integral del laboratorio remoto. Los principales componentes instalados en el servidor son los siguientes:

- ***DE2 web Server***. Un servidor web, implementado sobre Apache 2.2.14 con PHP 5.3.1, permite alojar el portal web que se encarga de la gestión de accesos y de la administración del laboratorio. Para tener acceso al laboratorio, los estudiantes deben registrarse y esperar a que el administrador acepte su petición. Los administradores disponen de un modo de auto-validación de lotes que permite automatizar la validación de todos los alumnos de un curso. Este laboratorio no contempla ni permite la experimentación simultánea de varios estudiantes. El primer estudiante que accede al laboratorio recibe una "licencia" reservando en exclusividad el acceso al hardware. La licencia expira después de 5 minutos, tiempo suficiente para probar el circuito, después de lo cual se acepta un nuevo inicio de sesión. Los usuarios que intentan iniciar sesión se les informa del tiempo restante cuando ya se está utilizando el hardware. Este monitoreo es una alternativa a un sistema de colas que aún no se implementa. Una vez el estudiante adquiere control sobre el laboratorio, puede acceder a la página de programación y seleccionar el archivo .sof que programará en la tarjeta de experimentación. Si la grabación es correcta, el usuario puede comenzar paralelamente la interacción con el experimento. El sistema incluye un panel web de administrador que permite, por ejemplo, ver quién es el usuario que está trabajando actualmente en el laboratorio y revocar la licencia de uso de este. El sistema permite la carga de páginas PHP o cualquier otro archivo, con fines de actualización, con efecto instantáneo. También hay un registro permanente para almacenar todos los eventos importantes ejecutados por los usuarios y el administrador. Cada base

de datos de este sistema es un archivo de texto simple. Las páginas web, programadas en PHP obtienen toda la información almacenada, como la lista de cuentas de usuario, mediante la lectura de archivos de texto

- ***Módulo de Programación***. Dado que el experimento está basado en una FPGA de Altera, el software encargado de su programación es el proporcionado por el propio fabricante. La Suite Quartus II carece de licencia abierta, pero es descargable gratuitamente y ofrece una utilidad para la grabación de dispositivos Altera desde línea de comandos. Esta utilidad es directamente lanzada desde la página PHP encargada de la grabación del sistema.

- ***Servidor de streaming de video***. Como se ha comentado anteriormente el streaming de la cámara se lleva a cabo mediante la aplicación Yawcam, es una aplicación Windows desarrollada en Java que proporciona streaming de video sobre un servidor web.

- ***Gestor de procesos***. Este componente permite la interacción con el experimento. Esta interacción se lleva a cabo utilizando el recurso "Editor del contenido de la memoria de sistema" que ofrece la aplicación Quartus II. Este recurso permite mediante comunicación a través del estándar JTAG editar los datos almacenados en la memoria de la tarjeta DE2 sobre la cual se implementa el experimento. Los circuitos electrónicos deben incluir un bloque virtual que se encarga de leer un conjunto de bits de memoria que representan a los pulsadores e interruptores con los que interactuar sobre los circuitos diseñados por los estudiantes. Este bloque se encarga de generar los niveles lógicos correspondientes a estas posiciones de memoria. El servidor de gestión ejecuta un programa residente desarrollado en C ++ que realiza llamadas a la consola de línea de comandos de Quartus II, en función del valor de un archivo de texto que contiene el valor de cada pulsador o interruptor. Una página PHP virtualiza los interruptores y pulsadores, aceptando las órdenes ejecutadas por los estudiantes desde la "caja de control" y escribe en el archivo de texto cualquier variación ejecutada (Figura 2.5). El programa C++ comprueba, con una frecuencia de 50Hz el estado del fichero y cuando detecta un cambio ejecuta los comandos oportunos directamente en la consola Quartus II SignalTap.

2.5.2.2 Cliente

Para poder trabajar con este laboratorio el cliente requiere dos ventanas independientes del navegador y una aplicación independiente (Figura 2.5).

1. Ventana del navegador con el DE2 webPortal para programar la FPGA del experimento cada vez que realiza una alteración en el circuito. Estas modificaciones sobre el circuito deben ser llevadas a cabo mediante la aplicación Quartus II.
2. Ventana del navegador con la página que muestra la captura de la webcam.
3. Aplicación Java que implementa la "caja de control".

2.5.2.3 Plataforma de experimentación

La plataforma encargada para llevar a cabo la experimentación es una tarjeta DE2, comercializada por Altera. El sistema de Desarrollo DE2 diseñado por Altera es una plataforma educativa para la experimentación con FPGAs. Constituye un vehículo ideal para el aprendizaje de la lógica digital, organización de computadores y FPGAs. Está fundamentada en una FPGA Altera Cyclone II 2C35 FPGA. Dispone de 8 MB de memoria SDRAM, 512 KB de memoria SRAM, y 4 MB de memoria Flash. Además proporciona múltiples periféricos de E/S como un LCD de 2 líneas por 16 caracteres, 18 interruptores, 4 pulsadores o 27 diodos led. Además, proporciona un nutrido conjunto de puertos de comunicación (USB 2,0, Ethernet, RS232, PS/2, IR, Audio In/Out, Video In/out, etc.) y conectores de expansión.

2.5.3 Escenario de uso

Este laboratorio remoto ha sido diseñado fundamentalmente para facilitar la experimentación de los estudiantes en el diseño de circuitos digitales. La elección de la plataforma de experimentación está basada en el software Quartus II, proporcionado por Altera que incluye un interfaz gráfico para el diseño de esquemáticos, que permite de manera sencilla realizar la captura de los mismos incluso por estudiantes sin conocimientos previos sobre circuitos digitales. Esta aplicación es gratuita y puede ser descargada libremente por los estudiantes e instalada en sus propios ordenadores, siempre y cuando dispongan estos de sistema operativo Windows Microsoft.

Para diseñar un circuito compatible con el laboratorio remoto DE2 web Portal, los estudiantes deben importar en la aplicación Quartus II un bloque virtual diseñado por los autores del laboratorio. Este bloque virtual se encarga de revisar el contenido de un conjunto de bits de memoria que contienen la información de las entradas para permitir la interacción con el experimento. El bloque virtual ofrece el estado de los 16 interruptores y los 4 pulsadores que el estudiante puede manipular desde la "Caja de control" (Figura 2.7). El archivo que contiene este bloque virtual, junto con instrucciones sobre su importación son descargables desde la propia web alojada en el servidor. Junto con el bloque virtual para el diseño de esquemáticos, los estudiantes pueden descargarse el código VHDL que implementa la misma función permitiendo el diseño de circuitos mediante este lenguaje de definición de hardware. Una vez diseñado el circuito los alumnos deben generar el fichero SRAM Object File (.sof) y acceder al portal web del laboratorio desde la dirección http://lsd.deec.uc.pt.

Una vez el alumno accede al portal, puede comprobar el estado del laboratorio, representado por una X verde si está libre o una X roja si está ocupada. Debido a que ningún sistema de colas o reserva está implementado, el alumno sólo puede observar el estado del laboratorio. Si el laboratorio está siendo usado por otro usuario, el estudiante deberá esperar a que se libere la sesión activa y el laboratorio quede libre. Una vez el estudiante se identifica, el laboratorio queda asignado y la X se transforma en una V verde (Figura 2.8).

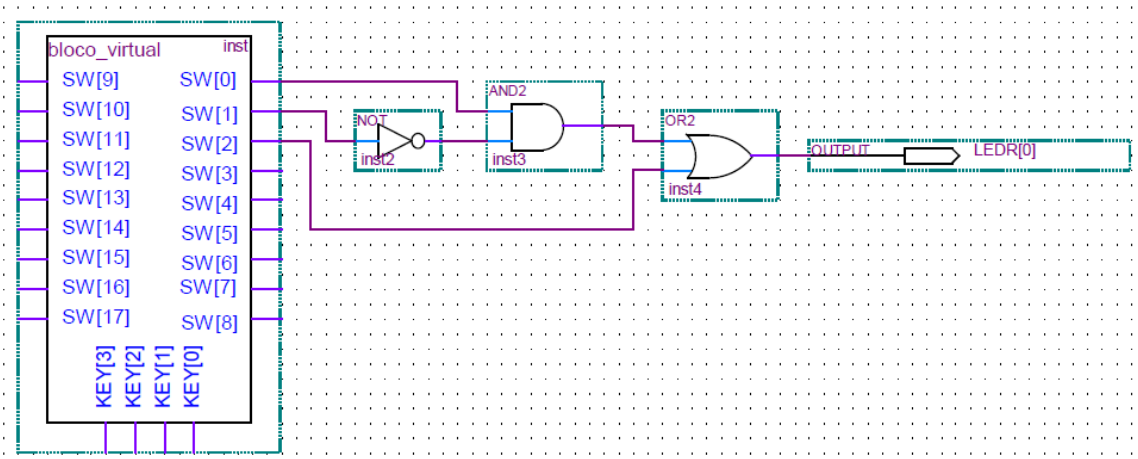

Figura 2.7. Esquemático diseñado por la aplicación Quartus2, incluyendo el bloque virtual requerido para la interacción con el laboratorio remoto DE2 web Portal.

Una vez el estudiante recibe la licencia de utilización del laboratorio y mientras esta permanezca activa (5 minutos) el usuario puede acceder a la página de subir circuito (UPLOAD E PROGRAMAR), abrir otra ventana del navegador con el applet que permite monitorizar la cámara (VER PLACA) y abrir una tercera ventana del navegador con la "Caja de control".

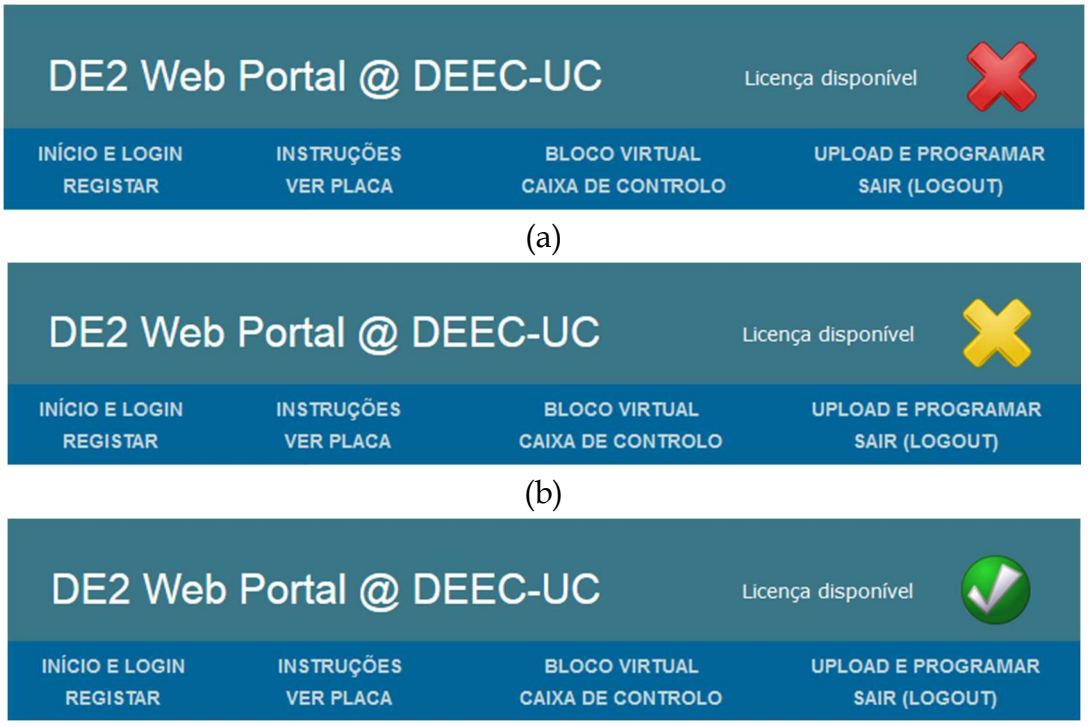

Figura 2.8. Sistema del laboratorio DE2 web Portal para comporbar el estado del experimento. (a) Experimento en uso por otro usuario, (b) experimento libre y (c) experimento asignado a usuario.

A partir de este momento el usuario dispone de tiempo para interactuar con el experimento a través de la caja de control. Este interfaz no permite alterar simultáneamente más de una entrada y requiere 1 segundo para llevar a cabo cada orden dada por el usuario, limitando las acciones de validación del sistema.

Mientras la licencia permanezca activa el usuario puede alterar el sistema mediante la aplicación Quartus II y reprogramar el experimento las veces que sean necesarias. Existe la posibilidad de acceder al administrador para solicitar un periodo de experimentación mayor.

2.5.4 Resumen

Este laboratorio remoto permite la experimentación con circuitos digitales diseñados y FPGAs utilizando la aplicación de captura de esquemáticos incluida en la suite Quartus II o mediante código VHDL.

La gran ventaja de este laboratorio es que, aunque no se trata de aplicaciones de código abierto, todos los componentes software utilizados para su despliegue son gratuitos y no requiere de ningún componente hardware adicional para el desarrollo de la experimentación remota. Al incluir el bloque virtual que posibilita la interacción con el laboratorio en el propio experimento diseñado por el estudiante, los únicos componentes que se necesitan para el desarrollo de este laboratorio remoto son:

- Un servidor con sistema operativo Microsoft Windows. Al no permitir accesos simultáneos al experimento, el rendimiento de este sistema no es crítico y puede ser implementado sobre cualquier computador de escritorio.
- La plataforma de experimentación DE2 de Altera.
- Una webcam USB.

De esta forma se limita el coste total del experimento a una cantidad inferior a 600 €.

En cuanto a la usabilidad, los principales problemas que presenta este sistema son los siguientes:

- El applet utilizado para visualizar la placa es muy antiguo, carece de firma y requiere deshabilitar la seguridad del navegador e instalar un driver propietario (Yawcam) para que funcione.
- La interacción con las entradas es muy lenta e impide la conmutación simultánea de varias entradas.
- El sistema de reserva es muy básico y al no tener implementado ningún sistema de colas o reservas puede resultar caótico cuando varios estudiantes quieran acceder simultáneamente al laboratorio.

2.6 "Vicilab" de la Universidad Nacional de Irlanda en Galway

Este laboratorio remoto está centrado en electrónica digital básica y FPGA, y ha sido desarrollado por el grupo de investigación en computación reconfigurable de la Universidad Nacional de Irlanda en Galway, dirigido por el Dr. Fearghal Morgan (Morgan & Cawley, 2011) (Kępa, y otros, 2010) (Morgan, y otros, 2011). En el año 2011 finaliza el desarrollo del "Remote FPGA Lab" o RFL, que ofrece un interfaz de visualización novedoso para la monitorización e interacción con una implementación hardware sobre una FPGA remota, proporcionando al estudiante una animación del comportamiento del sistema de lógica digital en cualquier nivel de la jerarquía de diseño (Morgan, Cawley, & Newell, 2012). En 2014 el RFL sufre una actualización a nivel hardware y software y es complementado con un nutrido conjunto de demostraciones interactivas en tiempo real que implementan sistemas digitales didácticos, todos ellos disponibles con un solo clic. Para gestionar este nuevo laboratorio "ViciLab" se creó una spin-off, denominada ViciLogic, que es liderada por el equipo original y participada por nuevos socios que aportan una mayor fiabilidad al sistema y mejor usabilidad (Morgan, y otros, 2014).

2.6.1 Funcionalidad

El RFL permite experimentar con el sistema de desarrollo bajo FPGA Nexys 3, fabricado y comercializado por Digilent (Digilent, 2013). Esta plataforma está basada en una FPGA Spartan 6 LX16 fabricada por Xilinx (Xilinx, 2015). Independientemente de la tecnología actual de experimentación, este laboratorio remoto presenta una arquitectura escalable y expandible que permite experimentar con varias plataformas embebidas.

2.6.1.1 Integración con otras plataformas de aprendizaje o experimentación

El laboratorio remoto Vicilab está integrado dentro de la plataforma de aprendizaje Vicilogic. Esta plataforma incluye recursos para el aprendizaje interactivo y evaluación en línea de sistemas digitales posibilitando, a través del propio laboratorio el diseño y desarrollo de prototipos digitales en la nube, y su evaluación interactiva mediante interfaces gráficos de usuario. Vicilogic incluye una plataforma online para la gestión del aprendizaje en sistemas digitales, denominada Vicilearn. Esta plataforma utiliza el laboratorio remoto para proporcionar experimentación con los contenidos didácticos. La integración entre ambos sistemas está desarrollada de forma natural y con un simple clic permite acceder a los interfaces gráficos de los circuitos incluidos.

Aparte de la integración de las dos herramientas mencionadas, los autores no han contemplado la integración de este laboratorio en ninguna plataforma de gestión de aprendizaje (LMS) ni sistema de gestión de laboratorios remotos (RLMS). En este sentido el desarrollo empresarial de la plataforma y su gestión por una empresa privada dificulta la federación con otras plataformas.

2.6.1.2 Escalabilidad

La escalabilidad es uno de los principales requisitos establecidos por los desarrolladores. Actualmente el laboratorio remoto dispone de 16 instancias de experimentación, todas ellas basadas en la plataforma de desarrollo Nexys 3 de Digilent (Figura 2.9).

Figura 2.9. Torres de plataformas de experimentación desplegadas para aportar escalabilidad al laboratorio Vicilab. Cada torre incluye 8 tarjetas Nexys 3 y un microservidor JTAG.

La comunicación con las plataformas de experimentación se lleva cabo mediante un microservidor JTAG conectado a la red local mediante conexión Ethernet. Cada microservidor JTAG permite la conexión de una matriz de hasta 8 instancias. La velocidad del protocolo Ethernet permite añadir nuevos microservidores al sistema mediante un conmutador de red (switch) sin alterar significativamente la latencia del sistema.

2.6.1.3 Desplegabilidad

Todos los componentes software y hardware utilizados para el despliegue del laboratorio son propiedad de Vicilogic. Actualmente no hay noticias del interés de esta empresa por el despliegue de nuevos laboratorios Vicilab en otras instalaciones, siendo este hecho complicado al fundamentar su modelo de negocio en la provisión de accesos al laboratorio mantenido en sus instalaciones.

2.6.1.4 Replicabilidad

Junto con la escalabilidad, la replicabilidad ha sido considerada un requisito fundamental en el desarrollo de este laboratorio remoto. En este sentido a finales del 2014 las plataformas de experimentación fueron actualizadas, sustituyendo las tarjetas Nexys 2 instaladas hasta ese momento, por nuevas tarjetas Nexys 3. Aunque actualmente todas las instancias de experimentación son idénticas, con el fin de facilitar

el balanceo de carga de los usuarios, la arquitectura es compatible con cualquier FPGA que soporte el core IP desarrollado para la comunicación con el servidor web del sistema mediante conexión Ethernet. Este core IP y los interfaces hardware con los dispositivos pueden ser adaptados a un importante rango de tecnologías FPGA. Actualmente varias plataformas basadas en FPGAs de Xilinx y Altera han sido probadas.

En cuanto a la replicabilidad del laboratorio para la experimentación con otras tecnologías de sistemas embebidos (MCUs, DSPs, etc…), es precisamente la exigencia de implementar un core IP desarrollado en VHDL lo que imposibilita cualquier adecuación posible.

2.6.1.5 Disponibilidad

Actualmente el acceso al laboratorio Vicilab es universal y gratuito. Cualquier usuario puede registrarse y acceder al laboratorio. El portal de Vicilogic incluye numerosa documentación y ejemplos que facilitan el acceso al laboratorio. Un nuevo usuario puede proceder a su registro desde el portal de Vicilogic (http://vicilogic.com) y rellenando un simple cuestionario que no exige confirmación, puede comenzar a experimentar con sistemas digitales en unos pocos segundos.

Para facilitar el acceso a la experimentación, el Portal web de Vicilogic almacena una cookie que automatiza el acceso de los usuarios con una vigencia de 90 días.

Si una sesión permanece inactiva durante 90 segundos es liberada permitiendo su reserva a otro estudiante.

2.6.1.6 Conectividad

Una de las grandes ventajas de este laboratorio es que no requiere ninguna configuración extra para su funcionamiento. Simplemente con conectividad a Internet y haciendo uso del puerto 80 (HTTP) el usuario puede experimentar tanto con los ejemplos dispuestos en la plataforma Vicilearn, plataforma didáctica para sistemas digitales que utiliza el laboratorio remoto para las animaciones, como con sus propias implementaciones llevadas a cabo mediante VHDL (Vicilab).

2.6.1.7 Universalidad

La universalidad es uno de los puntos débiles de este laboratorio. Al requerir la ejecución de su propia aplicación "Vicilab", disponible únicamente para plataformas Microsoft Windows, el catálogo de dispositivos desde los cuales podemos acceder es limitado. Esta característica imposibilita el acceso desde dispositivos móviles o chromebooks, limitando su accesibilidad desde aulas educativas.

Además requiere la instalación de la aplicación Xilinx Plannahead©, con unos requisitos importantes (4Gb RAM y 5GB HDD). Por otro lado la comunicación entre la

herramienta cliente y el servidor se desarrolla en forma de paquetes gestionados mediante la herramienta WinPCAP que debe también ser instalada.

2.6.2 Implementación

Este laboratorio remoto presenta una novedosa arquitectura fundamentada en la búsqueda de escalabilidad. Puede observarse en la Figura 2.10.

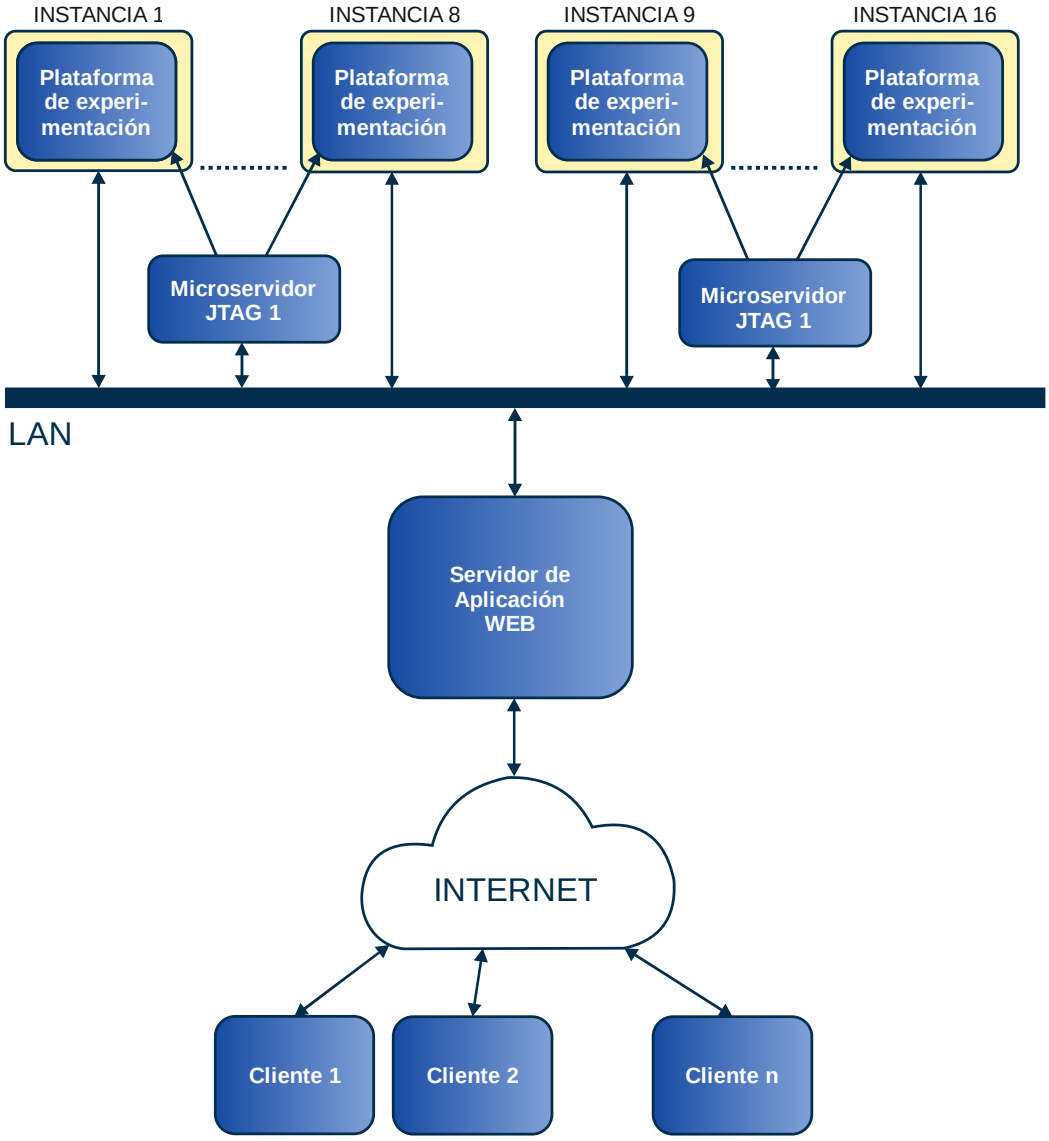

Figura 2.10. Arquitectura del laboratorio remoto ViciLab gestionado por Vicilogic.

2.6.2.1 Servidor de aplicación web

Este componente se encarga de gestionar las sesiones de usuarios, llevar a cabo la programación de las instancias de experimentación y generar los interfaces gráficos que serán consumidos por los estudiantes que permitirán la interacción con los experimentos.

Este componente software está desarrollado en Python y utiliza el framework Django para el desarrollo de las páginas web dinámicas. El servidor gestiona su propia base de datos de usuario, permitiendo el registro de nuevos usuarios y la gestión de los existentes por parte del administrador desde páginas web.

Este servidor soporta un conjunto de peticiones HTTP que permiten llevar a cabo toda la interacción con el servidor. Todas las consultas incluyen un identificador de sesión que garantiza la seguridad y permite identificar entre los múltiples usuarios simultáneos.

Los usuarios pueden y deben subir sus proyectos al servidor de Vicilogic, quedando siempre almacenados para su posterior apertura. Cada proyecto contiene tanto el fichero binario a programar en la FPGA de experimentación, como el interfaz gráfico de usuario que se ha diseñado para llevar a cabo las animaciones con el laboratorio. Una vez el proyecto es cargado en el servidor, el usuario puede solicitar mediante una petición HTTP la captura de la animación asociada. En ese momento el servidor programa la plataforma de experimentación asociada a la sesión y devuelve mediante HTML5 la web con el resultado del experimento correspondiente que, mediante AJAX, se recarga periódicamente generando una fluida animación. Desde esta web, el usuario puede pulsar sobre los componentes interactivos (interruptores y pulsadores), generando nuevas consultas HTTP que son gestionadas por el servidor.

El servidor de Aplicación web no dispone de conexión directa con el experimento. La programación de las plataformas de experimentación (FPGA) se lleva a cabo mediante los microservidores JTAG que permiten la conexión de dispositivos mediante el estándar Boundary Scan, recibiendo directamente el bitstream de cada implementación. Como se explicará en el siguiente apartado, la ejecución de los comandos de interacción es directamente gestionada por el core IP incluido en el fichero binario que ha sido implementado por la FPGA.

2.6.2.2 Plataforma de experimentación

Sin duda el propio sistema en el que se desarrolla el experimento es el componente fundamental de este laboratorio remoto. El enorme rendimiento de las FPGA de última o penúltima generación permite implementar sobre ellas complejos diseños de tal manera que los experimentos desarrollados por los estudiantes difícilmente requieren un pequeño porcentaje de las celdas integradas. Aprovechando esta circunstancia los desarrolladores han diseñado un core IP que envuelve el diseño del estudiante y se encarga de toda la comunicación entre el experimento diseñado del estudiante, el servidor de aplicación web y los propios periféricos de salida.

Este concepto, que en sí no es novedoso (Soares & Lobo, 2011), ha sido mejorado para optimizar la interacción con el laboratorio. La Figura 2.11 muestra el esquema del core IP que posibilita la interacción con el experimento. Puede observarse que se trata de un diseño jerárquico en cuyo nivel inferior se encuentra embebido el circuito implementado por el estudiante mediante VHDL. Todas las entradas, salidas e incluso

las señales entre los diferentes procesos del diseño que acapara el experimento son conectadas con registros de desplazamiento en cascada (CSR) que pueden ser consultadas desde capas superiores.

El mantenimiento y consumo de estos registros CSR constituye el nivel intermedio del núcleo. Además este nivel proporciona al estudiante recursos como memorias SRAM y SDRAM y acceso a los periféricos. La sustitución de la webcam como recurso de monitorización por la animación diseñada por el estudiante resta funcionalidad al uso de periféricos reales, aunque sigue siendo posible. Fundamentalmente existen dos registros CSR:

- SysProbe: Encargado de contener la información que circula por las señales compartidas por los procesos del experimento.
- UserControl: Este registro contiene el valor de cada entrada y salida del sistema.

Un nivel superior implementa los recursos de almacenamiento accesibles desde el nivel intermedio (Memorias SRAM y SDRAM) y la conectividad Ethernet que permite el acceso a los registros CSR a través de la red local posibilitando la interacción con el experimento.

La concurrencia exigida por el núcleo desarrollado imposibilita actualmente la portabilidad de esta solución a otras plataformas.

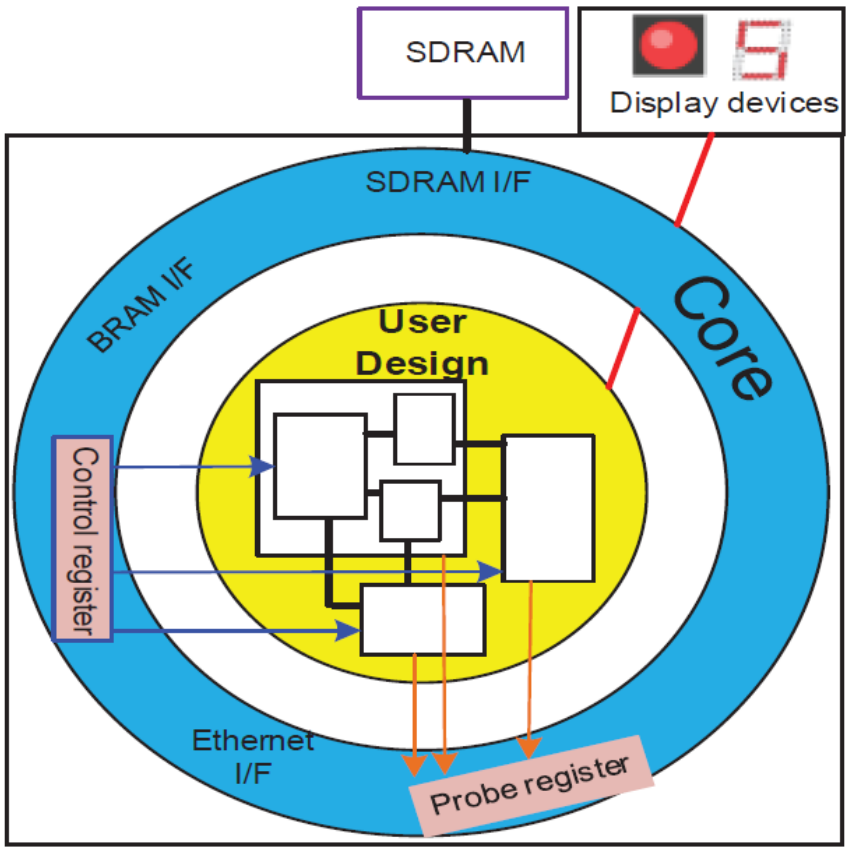

Figura 2.11. Núcleo Vicilogic que incluye al propio experimento diseñado por el estudiante y los componentes que permiten la interacción con el experimento.

2.6.2.3 Cliente

Como se ha comentado anteriormente este laboratorio remoto ha sido desarrollado para soportar dos tipos de experimentación.

Vicilearn

El objetivo de esta plataforma es posibilitar a educadores y estudiantes llevar a cabo animaciones no simuladas sobre circuitos didácticos en el área de los sistemas digitales. Esta experimentación se lleva a cabo mediante el Portal web ViciLearn, el cual contiene una potente base de datos de circuitos que incluye contenidos didácticos (descripción funcional, tabla de la verdad, ecuación booleana, diseño lógico, etc…) y el proyecto (binario + animación) que permite experimentar "realmente" con el circuito.

Para acceder a la plataforma Vicelearn únicamente es necesario disponer conexión a Internet y un navegador HTML que soporte HTML5. Además el diseño del cliente es User Responsive, facilitando el acceso desde dispositivos móviles. La Figura 2.12 muestra una captura del portal Vicilearn.

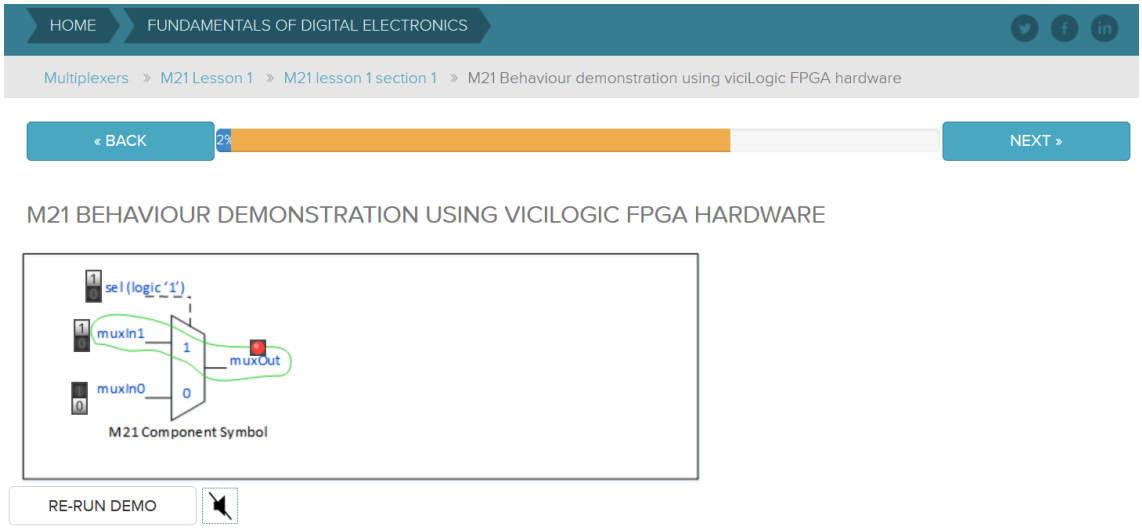

Figura 2.12. Captura del portal educativo para sistemas digitales ViciLearn.

Vicilab

Este laboratorio remoto permite evaluar sobre una plataforma de experimentación el funcionamiento de los circuitos implementados por los estudiantes mediante VHDL.

Toda la experimentación se desarrolla desde una aplicación cliente desarrollada para plataformas Microsoft Windows. Esta aplicación de escritorio permite establecer el modo de interactuación sobre las entradas y salidas del sistema y definir un interfaz gráfico de usuario sobre el que el usuario ejecutará la experimentación (Figura 2.13).

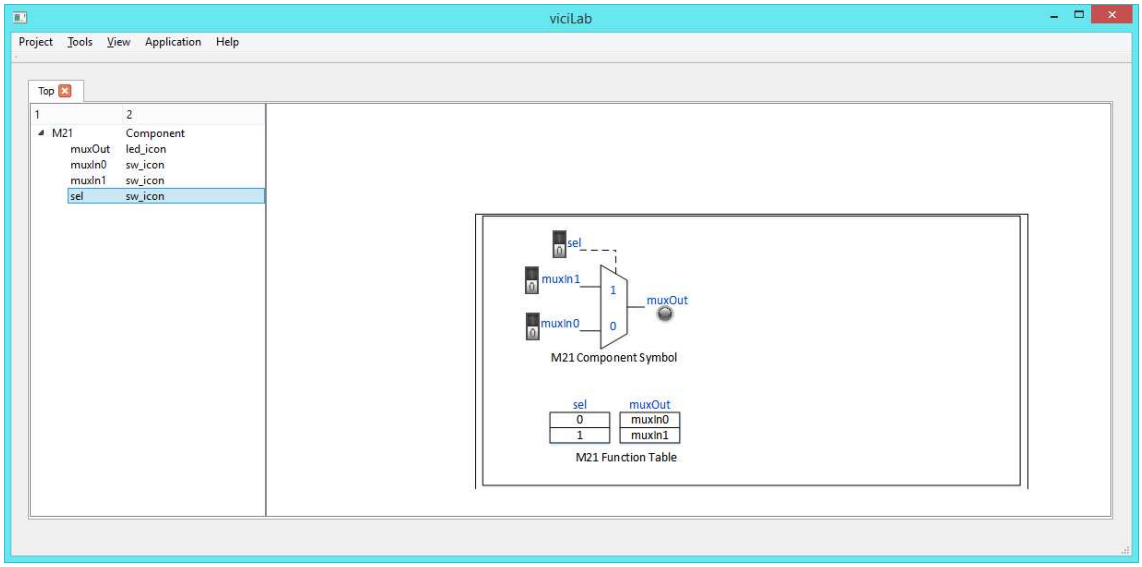

Figura 2.13. Captura de la aplicación Vicilab, sobre la que se ejecuta la experimentación con el laboratorio remoto de Vicilogic.

Este cliente no permite la codificación del VHDL que implementa el circuito, que debe ser editado desde herramientas externas. El formato aceptado es el generado por la suite de Xilinx WebPack ISE, siendo conveniente utilizar el editor VHDL incluido en dicha suite. El proceso de sintetizado del circuito debe ejecutarse dentro de la aplicación cliente. La aplicación embebe el circuito diseñado por el estudiante en el núcleo jerárquico desarrollado por Vicilogic y desarrolla el sintetizado mediante la herramienta externa PlanAhead de Xilinx.

La herramienta cliente incluye una aplicación para el diseño de las animaciones, en las cuales cada componente permite ser sustituido por una imagen sobre la cual se pueden colocar los controles interactivos de entrada y de salida.

2.6.3 Escenario de uso

El usuario deberá disponer de un ordenador bajo plataforma Microsoft Windows e instalar la aplicación Vicilab, el software Xilinx Planahead y el framework WinPCAP para poder desarrollar cualquier experimentación en este laboratorio remoto.

Los pasos a llevar a cabo por el estudiante para desarrollar un experimento en este laboratorio son los siguientes:

1. Implementar el circuito mediante el lenguaje de definición de hardware VHDL. Este paso debe ser desarrollado en el Project Manager de Xilinx o en cualquier editor de texto.
2. Una vez codificado el circuito, el cliente debe entrar en la aplicación Vicilab. Esta aplicación solo está disponible para plataformas Microsoft Windows e incorpora un interfaz gráfico que permite adaptar el circuito para su experimentación en el laboratorio remoto.

3. El primer paso es generar un nuevo proyecto. Cada proyecto debe tener un nombre diferente ya que todos ellos serán almacenados en el servidor, y este no permite la existencia de dos proyectos con el mismo nombre. Los proyectos subidos no se pueden borrar.
4. Importar el código VHDL. Esta herramienta soporta implementaciones en múltiples ficheros fuente, recibiendo como entrada la carpeta que contiene los ficheros .vhd del proyecto.
5. Una vez importados los ficheros fuente, la aplicación Vicilab reconoce todos los componentes implementados y las señales incluidas en cada uno de ellos. El siguiente paso es establecer las conexiones entre las señales y los controles (diodos, interruptores, pulsadores, Displays de 7 segmentos, etc…) que posteriormente nos permitan interactuar en la animación.
6. El siguiente paso es generar el fichero bitstream para la FPGA. El software ViciLab, integra la implementación del estudiante en su propio nucleo y sintetiza el bloque completo en un fichero bit. El sintetizado se lleva a cabo mediante la aplicación Xilinx PlanAhead que el estudiante deberá tener instalada en su ordenador personal. La vinculación entre ambos sistemas debe ser efectuada editando un fichero de configuración.
7. Posteriormente el usuario deberá generar un interfaz gráfico de usuario. Para ello la aplicación Vicilab dispone de una herramienta que permite disponer una imagen para representar a cada componente y disponer sobre esta todas las señales conectadas a dispositivos de monitorización o control.
8. Generados el binario y la interfaz de usuario, el estudiante debe llevar a cabo el "upload" del proyecto. La nube de Vicilogic conserva todos los proyectos almacenados por cada usuario por lo que ante cada modificación será necesario modificar el nombre del proyecto, en las propiedades del mismo.
9. A partir de este momento el usuario podrá ejecutar la animación y comprobar el funcionamiento del sistema diseñado.

2.6.4 Resumen

Este laboratorio supone una herramienta didáctica de primer nivel para el estudio de sistemas digitales. El portal Vicilearn dispone de contenidos didácticos de calidad incluyendo para cada circuito lógico información sobre todas las etapas y niveles del diseño (Bloque funcional, tablas de verdad, diagramas de Karnaugh, Ecuación booleana, circuito basado en puertas, etc…) y además permite experimentar con el circuito a través de una animación que se desarrolla sobre un potente interfaz gráfico, basado en una implementación que se ejecuta sobre una FPGA remota.

Sin embargo el proceso para experimentar con diseños propios resulta complicado. Es cierto que la aplicación se encuentra en una de sus primeras versiones (v1.3), pero su usabilidad es muy limitada. Aunque los pasos a desarrollar se encuentran documentados, el proceso es complejo y difícilmente asimilable por los estudiantes de

los primeros cursos. Los problemas y errores que aparece durante la ejecución de la aplicación no reportan suficiente información causando malestar en el estudiante.

En cuanto a la arquitectura del laboratorio remoto, las dos grandes cualidades son la escalabilidad y la replicabilidad sobre diferentes plataformas basadas en FPGA.

Por otro lado, es uno de los pocos laboratorios remotos para el desarrollo de sistemas embebidos que elimina el uso de una webcam para monitorizar la plataforma de experimentación. El resultado carece de inmersión, dado que por muy visual que se diseñe el interfaz gráfico de usuario, la sensación del estudiante no difiere de cualquier simulación (Ma, 2006) (Corter, y otros, 2007) (Corter, Nickerson, Esche, & Chassapis, 2004).

Su utilización requiere de tiempo de aprendizaje y para la experimentación requiere ejecutar numerosos pasos específicos totalmente carentes de sentido en un laboratorio presencial. Si uno de los objetivos de un laboratorio remoto debe ser permitir llevar a cabo una experimentación a distancia similar a la desarrollada presencialmente en un laboratorio, este sistema no lo cumple en absoluto.

2.7 Labshare FPGA Rig de The Labshare Institute

Este laboratorio forma parte de la oferta de experimentos remotos disponible por "The Labshare Institute (TLI)" en Australia. Esta organización sin ánimo de lucro nació para garantizar la sostenibilidad en el aprovechamiento de la utilización de las tecnologías de laboratorio de acceso remoto en el sector de la educación (Lindsay, Liu, Murray, & Lowe, 2007) (Murray, Lowe, Lindsay, Lasky, & Liu, 2008) (Lowe, Murray, Lindsay, & Liu, 2009). Su función principal es la de actuar como intermediario entre socios proveedores de laboratorios remotos y socios consumidores, siendo la responsable de gestionar el acceso y los recursos asociados a los laboratorios remotos entre las organizaciones asociadas que se han suscrito a los servicios de TLI.

2.7.1 Funcionalidad

El laboratorio para experimentación con FPGA es uno de los laboratorios desarrollados y gestionados directamente por el TLI en sus instalaciones, dentro de la Universidad Tecnológica de Sidney (UTS). Su finalidad es permitir experimentar con FPGAs implementando circuitos basados en máquinas de estado como temporizadores, controlador de semáforos y componentes de los microprocesadores, como Unidades aritmético lógicas (ALU) simples. Actualmente el laboratorio no se encuentra disponible ya que se encuentra en proceso de actualización durante el cual las plataformas de experimentación originales basadas en el modelo de FPGA Spartan 2 fabricada por Xilinx, están siendo sustituidas por otras más avanzadas, basadas en una FPGA Spartan 6 del mismo fabricante.

2.7.1.1 Integración con otras plataformas de aprendizaje o experimentación

Una de las principales características de todos los laboratorios remotos soportados por el TLI es su integrabilidad con plataformas de gestión del aprendizaje (Moodle, Sakai, etc…) y las principales plataformas de gestión de laboratorios remotos (iLab, webLab Deusto, etc…). Labshare dispone de su propio sistema de gestión de laboratorios remotos (RLMS) denominado "Labshare Sahara". Este RLMS dispone desde su versión v2.0 de un componente, "LabConnector", específicamente desarrollado para facilitar su integración con otros sistemas LMS o RLMS. En su versión actual v3.3, este componente ha sido mejorado y se han desarrollado implementaciones para su integración con otros sistemas RLMS como el iLab del Massachusetts Institute of Technology (MIT) (Herbert Yeung, 2010), webLab Deusto (Orduna, Rodriguez-Gil, & Lopez-de-Ipina, 2012) o las plataformas para la compartición de laboratorios remotos Library of Labs (LiLa) (Richter, Tetour, & Boehringer, 2011) o Lab2Go (Zutin, Auer, Maier, & Niederstatter, 2010)

2.7.1.2 Escalabilidad

Todas las funciones que permiten la gestión de múltiples instancias de experimentación se encuentran implementadas en el propio RLMS Labshare Sahara. La plataforma de experimentación utilizada por este laboratorio es la tarjeta Nexys 3 de Digilent. La programación de esta tarjeta se lleva a cabo mediante el software Digilent Adept (Figura 2.14) por medio de un cable USB conectado al servidor del experimento. Este sistema de programación impide la conexión de más de una tarjeta simultáneamente al servidor del experimento, exigiendo por tanto un nuevo servidor por cada nueva instancia. Aunque Labshare Sahara dispone de recursos para el balanceo de carga entre instancias gestionadas desde diferentes servidores de experimentos, este hecho supone un encarecimiento importante del coste del laboratorio.

Figura 2.14. Instancia del laboratorio remoto Labshare FPGA Rig gestionada directamente por el TLI.

2.7.1.3 Desplegabilidad

Desde el punto de vista software, todos los componentes software del RLMS Labshare Sahara están publicados bajo licencia BSD y los autores proporcionan documentación para facilitar la instalación en otras instituciones. Aunque las configuraciones del cliente y servidor de experimento del laboratorio Labshare FPGA Rig no son públicas, la documentación suministrada incluye un tutorial para la creación de un nuevo Rig (Experimento).

Respecto al hardware, como ya se ha indicado anteriormente, la plataforma fundamental sobre la que se lleva a cabo la experimentación es una tarjeta comercial de bajo coste, si bien a esta se le ha conectado una tarjeta diseñada a medida que contiene los periféricos a gestionar durante las sesiones de los estudiantes. Los ficheros que contienen el esquemático y el layout de esta tarjeta no son públicos, complicando su replicabilidad en otras instituciones.

Toda la información referente específicamente al laboratorio remoto Labshare FPGA Rig hace alusión al consumo de este por parte de estudiantes, no existiendo ninguna documentación que se refiera al despliegue o arquitectura del laboratorio.

2.7.1.4 Replicabilidad

Respecto a la posibilidad de replicar este laboratorio sobre otras plataformas de experimentación, el hecho de que actualmente se encuentre en proceso de actualización motivado por el cambio hacia una plataforma más moderna, confirma su capacidad para adecuarse a nuevas tecnologías.

Sin embargo existen algunas limitaciones que complican cambiar la plataforma que sustenta la experimentación. En primer lugar la programación de la FPGA se lleva a cabo mediante el software Digilent Adept. Este software permite su ejecución desde línea de comandos facilitando su ejecución desde otras aplicaciones pero solo es compatible con algunos dispositivos FPGA fabricados con Xilinx de las series Spartan y Virtex.

Este laboratorio incluye dos tarjetas de circuito impreso fabricadas para la conexión de los periféricos de entrada y salida respectivamente, que han sido diseñadas a medida para la plataforma Digilent Nexys 3 y requieren ser modificadas para su adecuación a nuevas plataformas.

Finalmente, la fabricación de la nueva plataforma de experimentación escogida Nexys 3 ha sido descontinuada por el fabricante (Digilent), no siendo recomendable para nuevas instalaciones puesto que su disponibilidad se encuentra sujeta al stock actual. Este hecho complica el mantenimiento y sostenibilidad del propio laboratorio remoto.

2.7.1.5 Disponibilidad

Los laboratorios remotos gestionados por TLI están disponibles para todas las instituciones asociadas. Como se ha indicado anteriormente, durante la última revisión de este apartado el experimento no estaba disponible al encontrarse en plena actualización. En el año 2012 Xilinx declaró obsoletos todos los modelos pertenecientes a la familia de FPGA Spartan 2, por lo que las nuevas versiones del software de desarrollo no incorporan compatibilidad con el modelo utilizado en la primera versión del Labshare FPGA Rig, requiriendo para la experimentación la instalación de versiones antiguas e incompatibles con los nuevos sistemas operativos.

El RLMS Sahara de Labshare dispone de recursos para la gestión de usuarios, que únicamente necesitan disponer de una cuenta de usuario institucional de un centro asociado al TLI, con la que directamente pueden acceder al laboratorio. El proceso de autenticación puede ser configurado para cada instalación. Labshare Sahara dispone de balanceo de carga entre instancias idénticas o la posibilidad de seleccionar una determinada instancia por el estudiante, incluyendo la gestión de las colas con prioridades. Además incluye un sistema de reserva que garantiza la disponibilidad de una instancia para un estudiante en momento dado.

Aunque Labshare Sahara permite la publicación de experimentos interactivos o de ejecución remota por lotes, el Labshare FPGA Rig está solo disponible en modo interactivo, permitiendo al estudiante alterar el valor de las entradas (Figura 2.15).

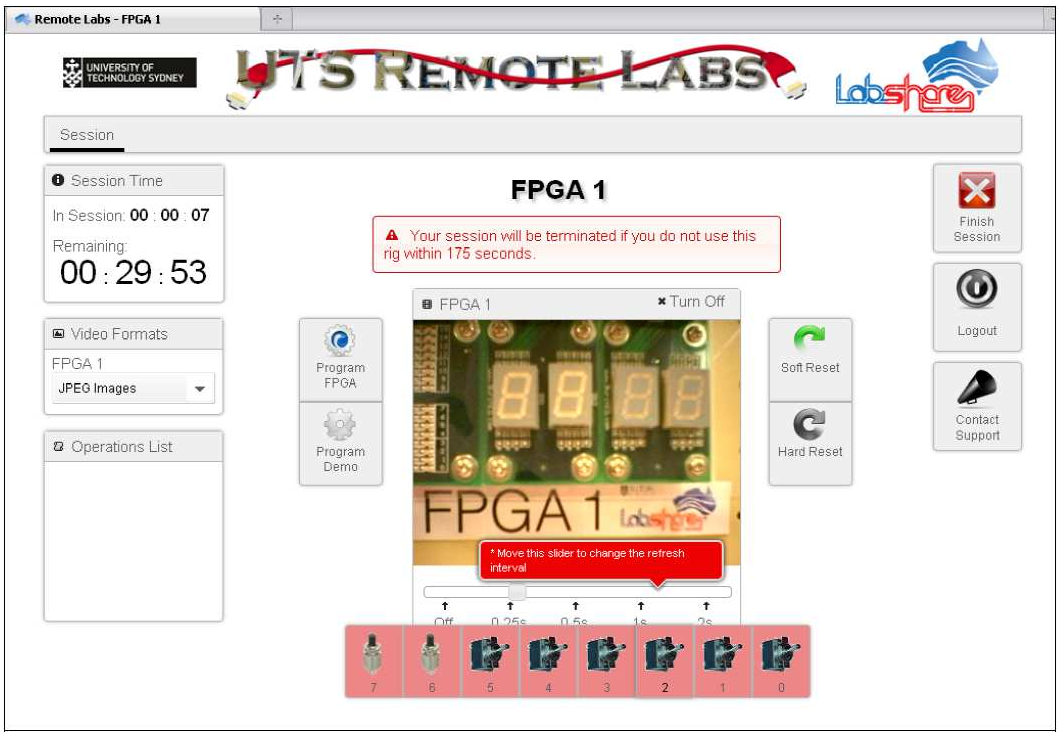

Figura 2.15. Sesión de experimentación en el laboratorio remoto Labshare FPGA Rig.

2.7.1.6 Conectividad

A través del RLMS, el usuario se conecta a través de un servidor web, que le permite llevar a cabo toda la experimentación desde un navegador (Figura 2.15).

2.7.1.7 Universalidad

Cualquier dispositivo que incluya un navegador web y conexión a Internet puede experimentar con este laboratorio remoto (Figura 2.15).

Sin embargo, el desarrollo de los experimentos por los estudiantes debe ser llevado a cabo bajo la suite Xilinx Webpack ISE. Esta plataforma está disponible para ordenadores con sistemas operativos Linux o Microsoft Windows.

2.7.2 Implementación

La implementación de este laboratorio viene marcada por las recomendaciones llevadas a cabo por los desarrolladores del RLMS Labshare Sahara (Lowe, Murray, Lindsay, & Liu, 2009) (Tawfik, 2013). La Figura 2.16 muestra la arquitectura básica del laboratorio. Los componentes fundamentales son los siguientes.

2.7.2.1 Servidor web

Soportado por un servidor Apache está desarrollado en HTML, PHP y Javascript. Su función es proporcionar al estudiante el interfaz gráfico de usuario que le permite desarrollar la experimentación. En Labshare Sahara, el Servidor web es el único componente directamente conectado a Internet y la única puerta de enlace con los estudiantes. Esto garantiza la universalidad de los dispositivos cliente y elimina cualquier problema de conexión causado por las infraestructuras de red.

2.7.2.2 Servidor de reserva

Este componente es el corazón de cualquier laboratorio remoto basado en el RLMS Labshare Sahara. Ejecuta todas las tareas administrativas, tales como la gestión de sesiones de usuarios, autenticación y por supuesto la reserva de las plataformas de experimentación. Dispone de un panel de control desde el cual los administradores pueden configurar permisos para usuarios y grupos, prioridades, tiempos de acceso y observar los accesos llevados a cabo a cada laboratorio o plataforma específica de experimentación. Está desarrollado en Java y dispone de un componente que permite balancear la carga entre instancias similares. Permite a los estudiantes solicitar una instancia cualquiera de un determinado laboratorio remoto o una específica, gestionando tanto colas como reservas.

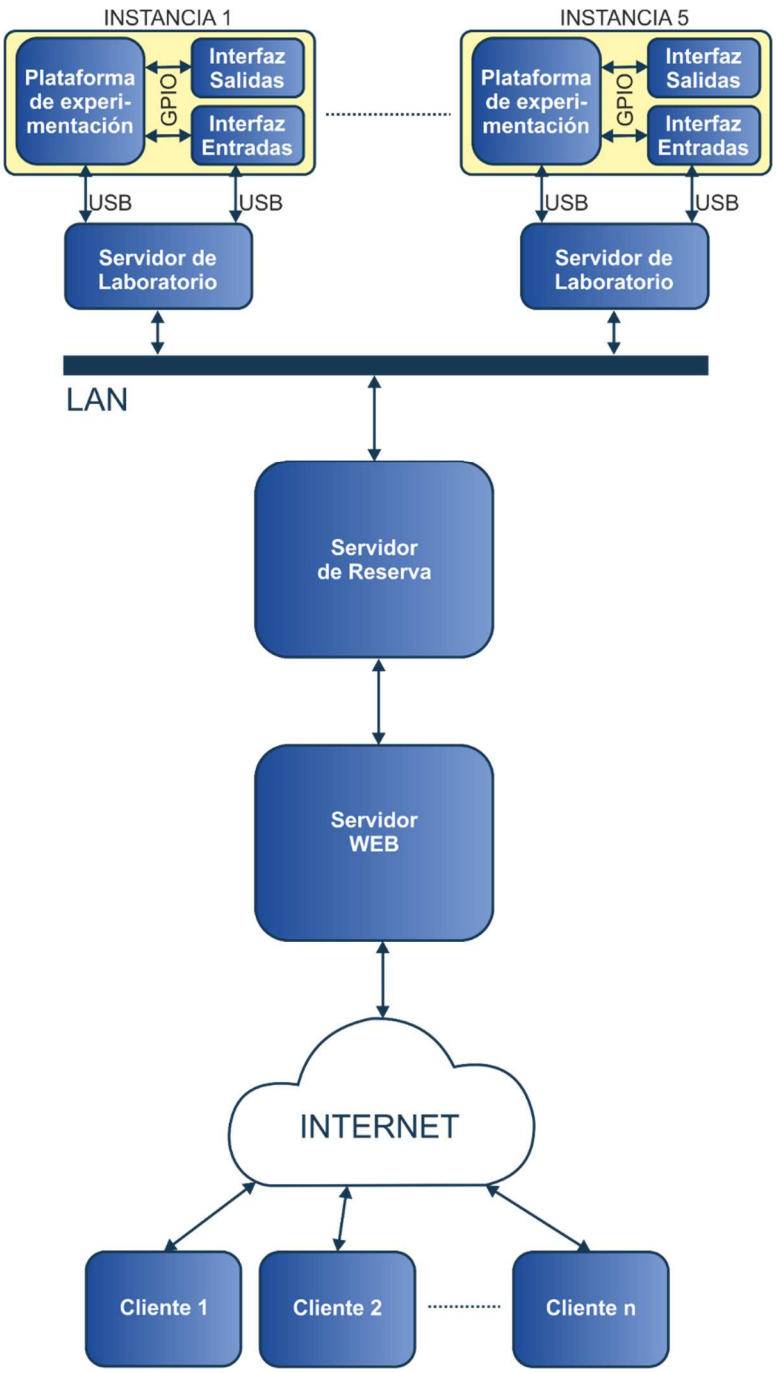

Figura 2.16. Arquitectura del laboratorio remoto Labshare FPGA Rig v.2.0.

2.7.2.3 Servidor de laboratorio

Contiene el software que controla directamente la plataforma de experimentación y los dispositivos encargados de interactuar con esta. Labshare Sahara permite la virtualización de los servidores de laboratorio permitiendo el despliegue de todos los servidores de laboratorio que controlan las múltiples instancias en una misma máquina. En el laboratorio remoto Labshare FPGA Rig v2.0, cada instancia de experimentación requiere su propio servidor de laboratorio. Esto es así porque la programación de las plataformas de experimentación se lleva a cabo mediante el software Adept de Digilent

que utiliza la utilidad Digilent Port Communications (DCP) la cual no permite la conexión simultánea de más de una FPGA. Este software dispone de un kit de desarrollo de software (SDK) que proporciona una API que permite generar sus propias aplicaciones de transferencia de datos para tarjetas Digilent. Además permite detectar la configuración JTAG y ejecutar aplicaciones de pruebas. La aplicación Adept incluye un programa para la programación de FPGAs desde línea de comandos que se utiliza en este laboratorio.

La segunda función del servidor de laboratorio es enviar a la tarjeta de interfaz de entradas, mediante conexión USB, las ordenes que gobiernan el valor de las señales conectadas a la plataforma de experimentación.

2.7.2.4 Plataforma de experimentación

La plataforma de experimentación está compuesta por una tarjeta Digilent Nexys 3, orientada a una FPGA Spartan 6 XC6LX16-CS324 fabricada por Xilinx. Esta tarjeta incluye 16Mbytes de memoria PSRam (pseudo-static DRAM), 32 Mbytes de memoria PCM Flash (16Mbytes mediante interfaz de periféricos serie SPI), Conectividad Ethernet, USB, UART, VGA, 8 interruptores, 5 pulsadores, 4 Displays de 7 segmentos, 8 diodos led y múltiples conectores de expansión. Además incluye un puerto USB Digilent Adept, que permite la alimentación y programación de la FPGA.

2.7.2.5 Interfaz de entradas

Este componente está implementado mediante tarjeta PCB diseñada a medida que permite interactuar con las entradas conectadas a la FPGA desde el interfaz web. Está directamente conectada con el servidor de laboratorio a través de un puerto USB y a la plataforma de experimentación mediante el conector VHDC, conectado a 20 pines del Banco 0.

Esta tarjeta se encarga fundamentalmente de recibir órdenes desde el panel de interruptores y pulsadores del interfaz gráfico y generar las señales correspondientes que son recibidas por la FPGA.

2.7.2.6 Interfaz de salidas

Directamente enfocada por una webcam permite monitorizar el estado de las salidas conectadas desde la FPGA. Se conecta a la plataforma de experimentación a través de varios conectores de expansión e incluye los siguientes periféricos:

- 6 displays alfanuméricos.
- 4 bloques de 4 displays de 7 segmentos multiplexados.
- 4 Servomotores.
- Múltiples diodos led
- Salida VGA

2.7.3 Escenario de uso

Mediante el uso de herramientas de desarrollo, disponibles de forma gratuita, los estudiantes pueden implementar un diseño específico en sus propios ordenadores personales utilizando VHDL. Para experimentar con el diseño, deberán iniciar sesión en el laboratorio remoto Labshare FPGA Rig V2.0 y subir el fichero binario generado mediante una interfaz web que también permite la interacción con los diseños en tiempo real. Esta interacción se desarrolla a través de interruptores y pulsadores virtualizados en el navegador web, y la monitorización del desarrollo del experimento se ofrece mediante una cámara enfocada a los diodos led, displays de 7 segmentos o displays alfanuméricos. Este laboratorio ha sido diseñado para el desarrollo de experimentos como sistemas basados en máquinas de estados, temporizadores y contadores, simulaciones de controlador de semáforo, control de servomotores, componentes de texto y control de pantallas gráficas. El rendimiento de la FPGA utilizada permite desarrollar componentes de microprocesadores, tales como la Unidad aritmética de Lógica (ALU), registros de desplazamiento múltiples y unidades de coma flotante.

El procedimiento para utilizar el Labshare FPGA Rig es común para los laboratorios soportados por el RLMS Labshare Sahara. Los estudiantes deben conectarse y autenticarse en la plataforma Labshare mantenida por el TLI. Inicialmente tras el acceso los estudiantes pueden elegir entre los laboratorios remotos mantenidos por la plataforma, en los cuales disponen de permisos para el acceso. Una vez seleccionan el laboratorio para experimentación con FPGA, los estudiantes podrán seleccionar una instancia cualquiera o bien seleccionar la instancia específica sobre la que desean experimentar (Figura 2.15). Seleccionada la instancia, el estudiante deberá confirmar si desea unirse a la cola que gestiona el acceso a la instancia. Una vez llegue el turno del estudiante, accederá al interfaz web de a instancia (Figura 2.17) y podrá comenzar con la experimentación. El tiempo de experimentación para cada usuario depende de la prioridad del mismo y de si ha reservado la instancia durante un periodo concreto.

Figura 2.17. Instancia de experimentación del Labshare FPGA Rig v2.0.

El interfaz gráfico de usuario proporcionado por el servidor web permite programar la FPGA con el fichero bitstream generado mediante la suite Xilinx Webpack ISE, que debe

estar instalada en el computador del estudiante. Una vez programada la FPGA el estudiante podrá comprobar el desarrollo de su experimento interactuando con los interruptores y pulsadores virtualizados y monitorizando mediante la webcam el comportamiento del sistema.

2.7.4 Resumen

Aunque la experiencia de usuario de los estudiantes es muy satisfactoria en este laboratorio, existen ciertos problemas que no han sido resueltos en la versión 2 del laboratorio remoto.

El requisito de sintetizar el diseño mediante la aplicación Xilinx Webpack ISE, exige que el desarrollo disponga de un computador con sistema operativo Linux o Windows, haciendo inútil la compatibilidad del laboratorio con dispositivos móviles que ofrece el RLMS Labshare Sahara.

Además pese a tratarse de un laboratorio que actualmente está en desarrollo, la selección de la plataforma de experimentación ha sido muy desafortunada por dos motivos:

1. la propia plataforma de experimentación, pese a ser relativamente reciente (2014), ha dejado de fabricarse, comprometiendo la sostenibilidad del laboratorio y el software requerido para desarrollar la experimentación con dicha plataforma, Xilinx Webpack ISE ha sido discontinuado por el fabricante
2. La actual versión de este software y única compatible con la FPGA incluida en la plataforma de experimentación, ocupa más de 17 Gigabytes complicando la instalación y descarga en el computador personal del estudiante.

Finalmente, aunque el rendimiento del laboratorio es apropiado incluso para dominios grandes de estudiantes, su coste es muy alto debido al número de dispositivos requeridos para su despliegue. La exigencia de un servidor de laboratorio por instancia, así como las tarjetas de interfaz de entrada y salida que se conectan a la plataforma de experimentación encarecen enormemente la réplica de este laboratorio.

2.8 CPLD Hybrid Lab de la Universidad de Ciencias Aplicadas de Carintia (CUAS)

Este laboratorio remoto ha sido desarrollado por el centro de competencias en laboratorios online de la Universidad de Ciencias Aplicadas de Carintia en Austria dirigido por el doctor Michael E. Auer. Originalmente este laboratorio fue desarrollado en colaboración con la Universidad Técnica de Illmenau para ser utilizado en el Master Europeo en Tecnologías Remotas, impartido en colaboración por ambas universidades durante los años 2010 y 2011 y financiado por la Unión Europea bajo el programa

Erasmus (Pop, Zutin, Auer, Henke, & Wuttke, 2011) (Zutin, Andrade, Restivo, & Rodrigues Quintas, 2013).

2.8.1 Funcionalidad

Este laboratorio se utiliza en cursos de desarrollo y prototipado rápido de sistemas digitales. Los alumnos utilizan lenguajes de definición de hardware (VHDL, Verilog o AHDL) o una utilidad de diseño de diagramas esquemático de bloques para capturar el circuito e implementarlo sobre un CPLD de la familia MAX 2, fabricado por Altera.

2.8.1.1 Integración con otras plataformas de aprendizaje o experimentación

Aunque durante los primeros años de funcionamiento este laboratorio funcionaba de manera autónoma, actualmente se ha adaptado a la arquitectura de compartición iLab y forma parte del catálogo de experimentos ofertados por el proyecto "iLab Europe Network". Además este laboratorio ha sido incluido en el portal Lab2Go (Zutin, Auer, Maier, & Niederstatter, 2010) (Restivo & Cardoso, 2012) que facilita la compartición de recursos de experimentación remota entre diferentes entidades (Figura 2.18).

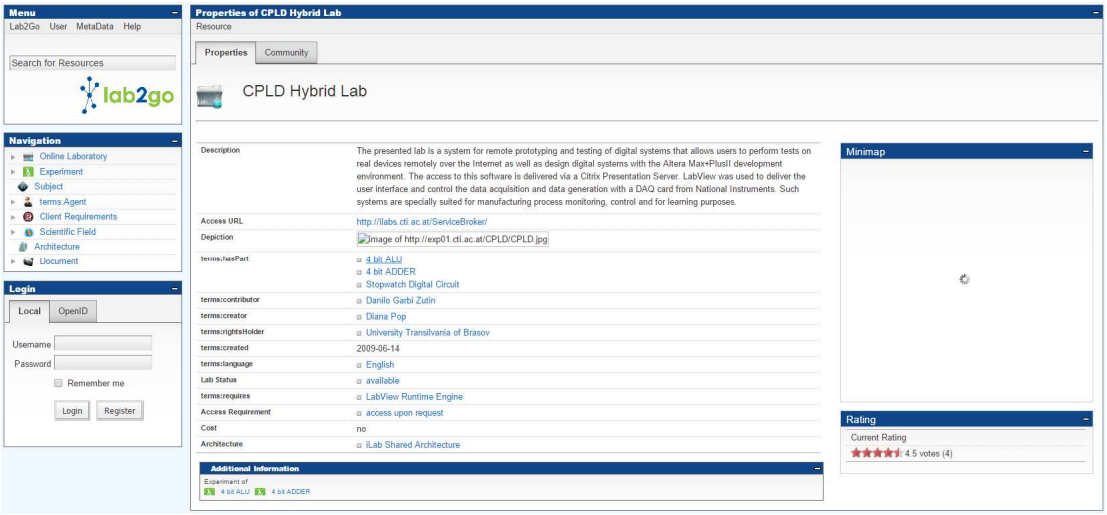

Figura 2.18. Información sobre el laboratorio CPLD Hybrid Lab publicada en el portal Lab2Go.

2.8.1.2 Escalabilidad

Este laboratorio remoto ha sido desarrollado para su utilización por grupos reducidos de estudiantes, no estando contemplada su escalabilidad.

La programación del dispositivo lógico programable sobre el que se implementa la experimentación (Altera MAX) se lleva a cabo desde un puerto paralelo que dificulta la conexión de nuevas instancias al servidor y lo limita a un máximo de dos.

2.8.1.3 Desplegabilidad

Tanto la plataforma de experimentación como la tarjeta de comunicación han sido diseñadas a medida para este laboratorio. No hay ninguna información publicada por los desarrolladores para el despliegue de nuevas instancias del laboratorio ni documentación abierta sobre el hardware o software del mismo.

Además las licencias de software requeridas, fundamentalmente las de LabView y Citrix, complican su portabilidad a otras instituciones educacionales o instalaciones.

2.8.1.4 Replicabilidad

En la misma línea que en los apartados anteriores, la total dependencia del laboratorio con el entorno de desarrollo MAX+PLUS2 de Altera, impide la adecuación de este laboratorio a otras familias de CPLD y por supuesto a otras tecnologías embebidas.

2.8.1.5 Disponibilidad

Como se ha comentado anteriormente este laboratorio está publicado en el portal "Lab2Go". Esto facilita su localización y proporciona información sobre el acceso al laboratorio y la experimentación posible. Este laboratorio se encuentra disponible para su acceso y en la modalidad de acceso bajo solicitud sin coste alguno.

2.8.1.6 Conectividad

En este apartado el único requisito impuesto por este laboratorio desde el punto de vista de la infraestructura de red de la instalación es la apertura del puerto TCP 135 para acceder al servidor de aplicaciones implementado mediante un servidor Citrix.

2.8.1.7 Universalidad

En el aspecto de la universalidad, el principal requisito de este laboratorio es el applet de Java necesario para el establecimiento de una sesión con el servidor Citrix que permite acceder al entorno de desarrollo remotamente. Lógicamente esto requiere un navegador compatible con applets y la instalación de máquina virtual de Java en cualquier dispositivo desde el que se quiera acceder al laboratorio. Aunque lamentablemente este hecho limita el número de dispositivos desde los que llevar a cabo la experimentación, la no necesidad de instalar ni ejecutar en el dispositivo cliente ningún entorno de desarrollo permite acceder desde dispositivos carentes de gran rendimiento.

Nuevas versiones de Citrix Server disponen de aplicaciones nativas para las principales plataformas móviles que permiten el acceso a las máquinas virtuales donde se ejecutan las aplicaciones compartidas. Sin embargo la versión empleada en el laboratorio no es compatible y requiere una actualización.

Adicionalmente la interacción con la plataforma de experimentación, se lleva cabo mediante un interfaz gráfico de usuario generado mediante LabView web Publishing. Por tanto la ejecución de este cliente requiere la instalación del plugin para LabView, no disponible para todos los navegadores y que presenta problemas en dispositivos con recursos de memoria limitado debido a su peso.

2.8.2 Implementación

La arquitectura de este laboratorio puede observarse en la Figura 2.19.

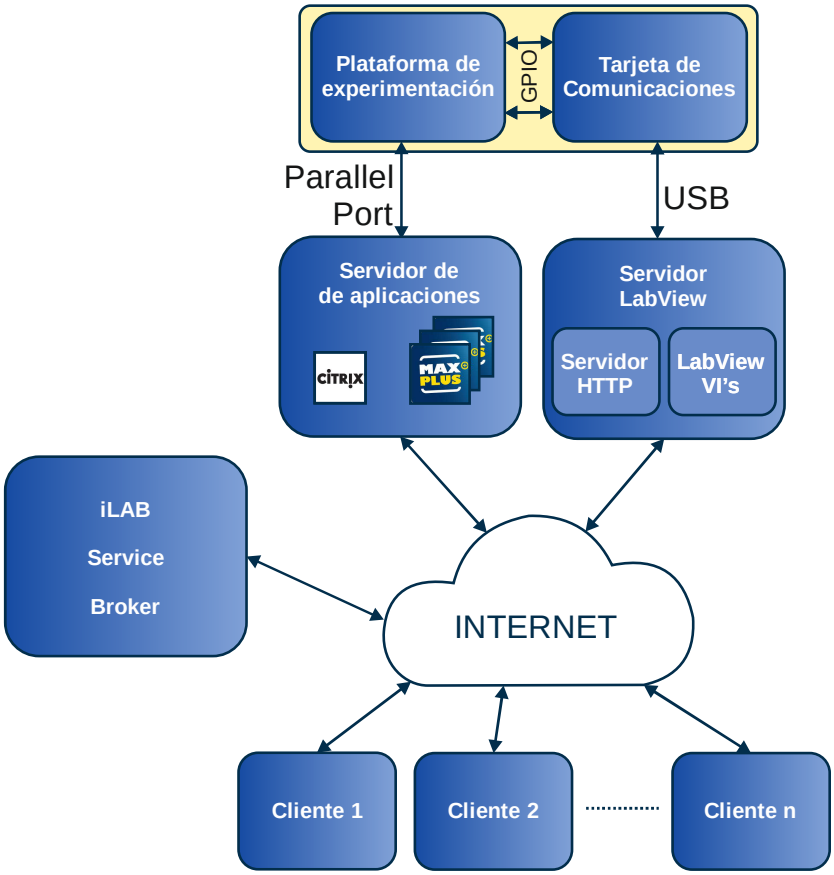

Figura 2.19. Arquitectura del laboratorio remoto CPLD Hybrid Lab de la Universidad de Ciencias Aplicadas de Carintia

2.8.2.1 Servidor de aplicaciones

El objetivo de este sistema es permitir al estudiante acceder remotamente al entorno de desarrollo MAX+PLUS II de Altera. Este componente está implementado mediante un servidor Citrix que permite virtualizar el entorno de desarrollo sobre un navegador a través de un applet de Java. Esta solución evita la necesidad de instalar el software de diseño sobre el cliente desde el que se acceder al laboratorio remoto y limita el rendimiento necesario del mismo.

Además permite al estudiante acceder a la plataforma de experimentación a través del puerto paralelo de forma idéntica a cómo lo haría en un laboratorio presencial.

2.8.2.2 Servidor LabView

Este sistema se encarga de proporcionar a los estudiantes un interfaz gráfico de usuario (GUI) que permita interactuar con las entradas de la plataforma de experimentación. Su implementación se ha llevado a cabo mediante LabView y utiliza la herramienta web Publishing para permitir el acceso remoto a través de un navegador compatible con el correspondiente plugin. Como se muestra en la Figura 2.20, el GUI permite virtualizar sobre una imagen de la plataforma el estado de los periféricos de salida (displays de 7 segmentos y diodos led).

Un diagrama de bloques diseñado mediante el LabView VI permite enviar comandos por el puerto de comunicaciones USB hacia la tarjeta de comunicaciones.

La utilización de Labview viene heredada de la primera versión de este laboratorio, en la que la tarjeta de comunicaciones estaba implementada sobre una tarjeta de entradas y salidas de National Instruments.

2.8.2.3 iLab Service Broker

La Arquitectura compartida iLab (ISA) es un framework software basado en servicios web que tiene como objetivo facilitar el acceso a los laboratorios en línea en base a características universales compartidas entre estos y servicios requeridos comunes. Distingue las tareas de utilización de un laboratorio específico que comprende un experimento de las tareas de gestión de cuentas de los usuarios, la autenticación de usuarios y otras tareas que registran una sesión de laboratorio. Esta separación de funciones es una de las principales ventajas de la arquitectura compartida iLab. ISA no se centra en un tipo específico de laboratorio, sino que proporciona un conjunto de funciones de uso general para los desarrolladores de laboratorio. ISA se divide en tres niveles que proporcionan diferentes servicios. Estos niveles son clientes, Service Broker y el servidor del laboratorio. El Service Broker es el núcleo de la arquitectura y proporciona autenticación de usuarios, la autorización, el almacenamiento de los datos del experimento y el acceso a servicios de programación.

2.8.2.4 Plataforma de experimentación

Como se ha mencionado anteriormente la plataforma de experimentación está basada en un CPLD de la familia Max, fabricado por Altera. El diseño de la tarjeta ha sido desarrollado a medida y permite la programación del CPLD a través de un puerto paralelo mediante la aplicación MAX+PLUS proporcionada gratuitamente por el fabricante Altera. Esta plataforma permite la experimentación remota a través del CPLD Hybrid Lab, pero también permite la experimentación convencional en un laboratorio

presencial (Figura 2.20). Los siguientes periféricos han sido incluidos para permitir la experimentación:

- 3 x display de 7 segmentos multiplexados
- 8 x diodos led
- 8 x conmutadores de desplazamiento
- 2 x pulsadores
- 1 x potenciómetro
- 1 x buzzer piezoeléctrico
- 1 x conector de expansión de 8 vías.

2.8.2.5 Tarjeta de comunicaciones

La tarjeta de comunicaciones ha sido implementada a medida para la implementación de este laboratorio remoto. Está basada en el microcontrolador PIC18F4550 que incluye conectividad USB 2.0 facilitando su conexión con el servidor de LabView. El firmware programado en el MCU recibe órdenes directamente desde el módulo de LabView VI que alteran el valor de 8 salidas digitales que son conectadas a través de un conector de 8 vías a la plataforma de experimentación para permitir la interacción en tiempo real.

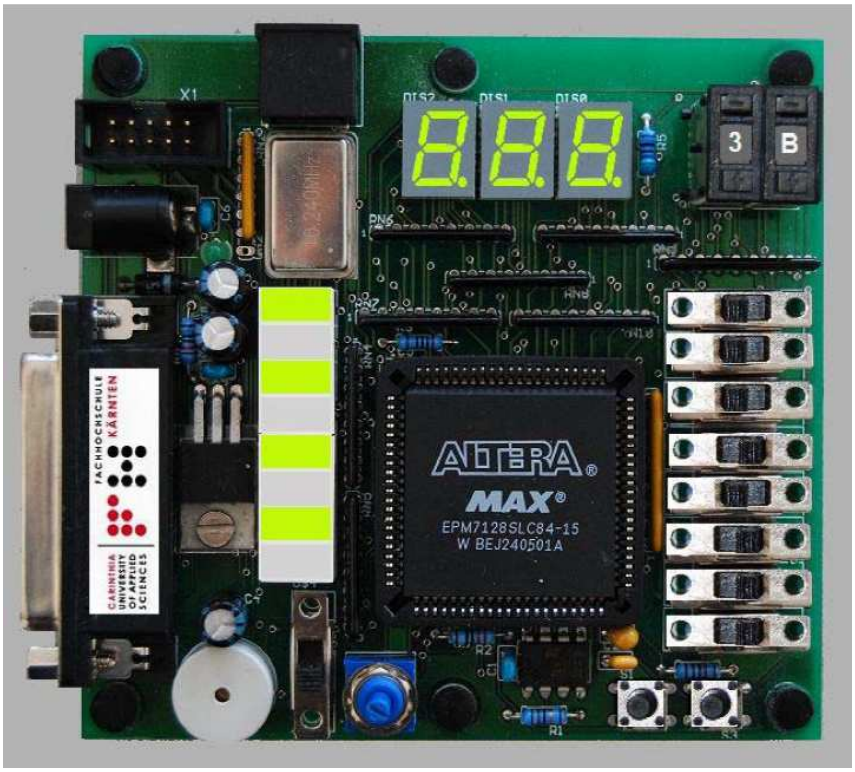

Figura 2.20. Interfaz gráfico de usuario que muestra la plataforma de experimentación utilizada en el CPLD Hybrid Lab

2.8.2.6 Cliente

La conexión del estudiante al laboratorio remoto se lleva a cabo a través de dos ventanas del navegador que permiten editar, sintetizar y programar la plataforma de

experimentación e interactuar en tiempo real con la implementación programada en el CPLD.

Ambas ventanas del navegador incluyen requisitos que impiden la universalidad del cliente. La ventana que nos permite acceder al entorno de desarrollo para CPLDs de Altera requiere la ejecución de un applet de Java que nos permite conectar con el servidor de aplicaciones implementado en un servidor Citrix y la ventana de interacción en tiempo real requiere la ejecución del plugin de LabView.

2.8.3 Escenario de uso

La experimentación en este laboratorio resulta muy similar a la desarrollada en un laboratorio presencial. Mediante el Service Broker Server de iLab el alumno debe autenticarse para acceder al laboratorio. El alumno dispone de un gestor de reservas que permite establecer intervalos de uso del laboratorio. Una vez el alumno recibe el control del laboratorio, puede conectarse con el servidor de aplicaciones, que a través de un navegador permite acceder a la aplicación MAX+PLUS de Altera.

Gracias al servidor Citrix, esta aplicación que se ejecuta sobre el navegador de forma idéntica a la que se ejecuta en cualquier ordenador de escritorio, permite la captura del circuito, la simulación, sintetizado, mapeo y programación de la plataforma de experimentación.

Una vez el estudiante recorre todo el flujo de diseño de sistemas sobre CPLD y la plataforma es correctamente programada con el firmware generado, aparece un link que permite abrir una nueva ventana del navegador que, mediante el interfaz gráfico de usuario, permite monitorizar los periféricos de salida e interactuar sobre los periféricos de entrada (Figura 2.20).

2.8.4 Resumen

Este laboratorio remoto resulta adecuado para la experimentación con pequeños grupos de estudiantes que no requieren un uso intensivo del mismo. Su principal característica que lo diferencia del resto es que permite al estudiante desarrollar un circuito mediante el entorno de desarrollo apropiado, sin necesidad de instalar dicha aplicación sobre la plataforma cliente. Esta misma característica conlleva unos tiempos de uso muy superiores a los del resto de laboratorios analizados, en los que todo el proceso de captura del circuito se desarrolla sin establecer conexión con el laboratorio. Esta característica unida al hecho de que la escalabilidad y la replicabilidad del laboratorio no están contempladas, impide el uso del mismo con dominios de estudiantes elevados.

Otra característica diferenciadora de este laboratorio remoto es la posibilidad de llevar a cabo simulación del circuito previamente a la implementación del mismo en el propio CPLD. La aplicación MAX+PLUS dispone de un simulador que permite validar

el circuito previamente a llevar a cabo la programación del mismo. Esta peculiar característica otorga el nombre al laboratorio al convertirlo en un laboratorio híbrido con capacidad Virtual y Real.

En cuanto a la implementación del laboratorio, el empleo de herramientas software con licencias de pago (Citrix y LabView) encarece enormemente el coste del mismo.

2.9 AT89S52 MCU Remote lab de la Universidad de Indonesia

Este laboratorio ha sido desarrollado por el Departamento de Ingeniería Eléctrica de la Universidad de Indonesia (Limpraptono, Ratna, & Sudibyo, 2012) (Limpraptono, Sudibyo, Ratna, & Arifin, 2011). El componente fundamental del laboratorio es un Microservidor web desarrollado para ejecutar la comunicación con la plataforma de experimentación.

2.9.1 Funcionalidad

Este laboratorio permite experimentar remotamente con un microcontrolador AT89S52, fabricado por Atmel. Los estudiantes pueden llevar a cabo la programación del dispositivo, interactuar con entradas digitales directamente conectadas al mismo y monitorizar, mediante una webcam, el resultado de la ejecución de su programa y el comportamiento de los periféricos de salida.

2.9.1.1 Integración con otras plataformas de aprendizaje o experimentación

En el desarrollo del laboratorio no se ha tenido en cuenta la integración del mismo en ninguna plataforma de aprendizaje ni sistema de gestión de laboratorios remotos. El hecho de que el interface gráfico de usuario que es consumido por el cliente sea suministrado directamente por el principal servidor web que a su vez ejecuta las tareas de administración complica su integración con otros sistemas.

2.9.1.2 Escalabilidad

Esta característica, junto con el coste global del sistema, ha sido el principal requisito tenido en cuenta en el desarrollo del laboratorio remoto. Dado que la programación e interacción con la plataforma de experimentación se lleva a cabo utilizando exclusivamente un microservidor web y la consiguiente webcam, ambas reconocibles a través de su dirección IP, el sistema soporta tantas instancias como se desee, existiendo como única limitación la propia capacidad de la infraestructura de red de la instalación sobre la que se despliegue (Figura 2.21).

Para cada nueva instancia, el servidor principal aloja una web independiente e idéntica que permite la reserva y gestión de la misma. Simplemente hay que replicar la

propia web y la base de datos de reservas implementada en mySQL. Un archivo de configuración permite indicar la dirección IP del microservidor y la webcam correspondiente a cada instancia, así como la cadena de conexión de la base de datos asociada a dicha instancia.

Figura 2.21. Fotografía de dos instancias del laboratorio remoto para microcontroladores de la Universidad de Indonesia.

2.9.1.3 Desplegabilidad

No existe ningún repositorio que contenga información sobre los componentes software y hardware requeridos para el despliegue de este laboratorio, siendo complicado su despliegue en otras instituciones.

Por otro lado la interconexión de todos los subsistemas a través de una red local mediante conexión Ethernet facilita enormemente el despliegue de los mismos permitiendo su disposición en distintos puntos de la instalación aprovechando la infraestructura de comunicaciones de la misma.

2.9.1.4 Replicabilidad

Debido a que el firmware del microservidor está desarrollado a medida para este modelo de microcontrolador (Atmel AT89S52), es imposible su adaptación a otras plataformas. Para ello es necesario que el sistema permita su reprogramación mediante un bootloader y reprogramar el componente de programación del microservidor adecuándolo a las características de la arquitectura de la nueva plataforma y del bootloader habilitado en la misma.

2.9.1.5 Disponibilidad

Para acceder al laboratorio es necesario disponer de unas credenciales que deben ser generadas por los administradores del sistema. Por otro lado el sistema únicamente se activa durante el desarrollo de ciertos cursos impartidos por la Universidad de Indonesia para las que representa un material didáctico fundamental.

Además, apenas existe material didáctico y ejemplos de uso, lo cual complica el desarrollo de experimentos en el laboratorio.

Finalmente el bootloader que permite la programación del microcontrolador principal que gobierna la plataforma de experimentación impide el acceso a múltiples registros especiales del mismo lo cual limita las posibilidades de experimentación.

2.9.1.6 Conectividad

Una de las grandes ventajas de este laboratorio es que no requiere ninguna configuración extra para su funcionamiento. Toda la comunicación del estudiante con el laboratorio se lleva a cabo a través de una página web.

2.9.1.7 Universalidad

El único requisito del dispositivo cliente para interactuar con el laboratorio es disponer de un navegador web.

Además como el firmware de cada experimento debe ser implementado en el dispositivo cliente, el estudiante requiere disponer de un entorno de desarrollo para el microcontrolador Atmel AT89S52. Aunque existen numerosos entornos de desarrollo integrados que soportan esta familia de microcontroladores, muy pocos permiten alterar las posiciones de los vectores de Reset e Interrupciones, algo necesario para generar un archivo binario compatible con el bootloader instalado en el microcontrolador principal de la plataforma de experimentación. Los dos recomendados por los desarrolladores, Keil y Atmel IDE, solo están disponibles para plataformas Microsoft Windows.

2.9.2 Implementación

El esquema de este laboratorio puede observarse en la Figura 2.22. Como se indica a continuación la particularidad principal de este laboratorio es la completa independencia de la plataforma de experimentación respecto al servidor principal. Esta característica es debida a que el microservidor se encarga directamente de gestionar tanto la programación como la interacción del experimento.

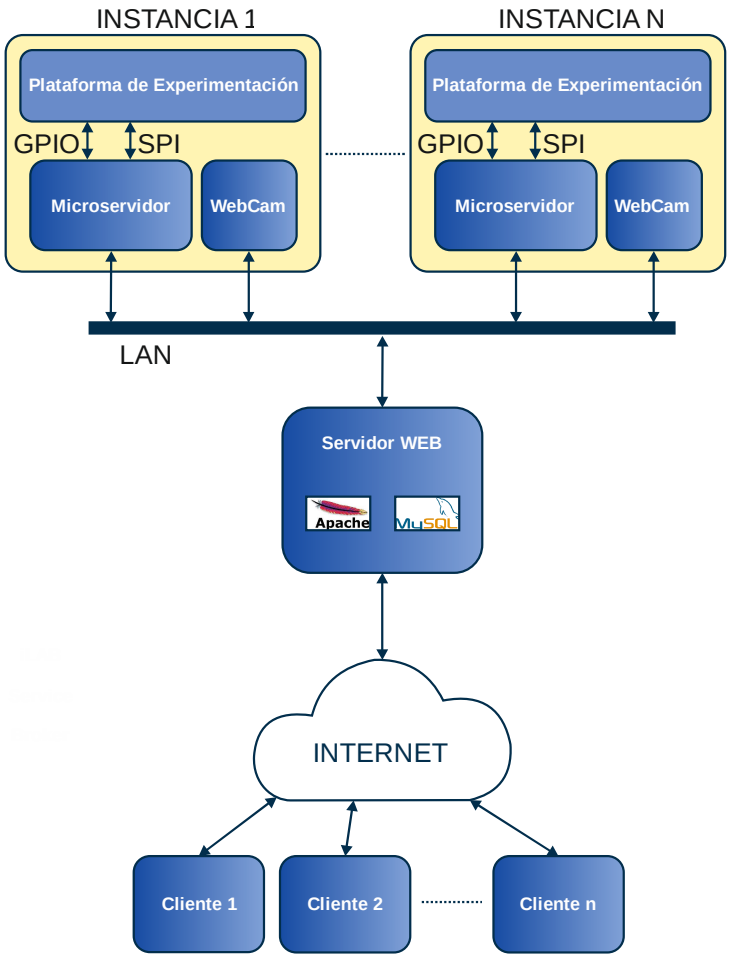

Figura 2.22. Arquitectura del laboratorio remoto para experimentación con microcontroladores de la Universidad de Indonesia.

2.9.2.1 Servidor web

Es el único subsistema del laboratorio conectado a Internet y lleva a cabo toda la interacción con los estudiantes. Dispone de un servidor web implementado sobre Apache que se encarga de proporcionar un portal genérico que incluye las tareas principales de administración como autenticación y gestión de sesiones. Una base de datos implementada en MySQL almacena la información de los usuarios y registra los accesos realizados. El portal principal muestra una instantánea de cada instancia activa indicando si en ese momento se encuentra ocupada o libre. Para cada instancia, el servidor aloja una web independiente que permite experimentar al estudiante. Cada web perteneciente a una instancia dispone de su propio sistema de reservas independiente. Una vez el estudiante reserva una instancia libre e inicia la sesión el navegador dispone de un interfaz gráfico de usuario (Figura 2.23), que permite subir el fichero binario, monitorizar la plataforma de experimentación desde una webcam e interactuar por medio de 4 entradas digitales que pueden ser alteradas en tiempo real. El servidor proporciona un chat que permite comunicar al usuario de una instancia con el administrador del sistema.

2.9.2.2 Microservidor

Como se ha comentado antes el microservidor es el componente fundamental del sistema. Se encarga de programar el microcontrolador principal de la plataforma de evaluación y permitir la interacción con la misma posibilitando la alteración en tiempo real de 4 entradas digitales conectadas al MCU.

Fundamentalmente está compuesta de dos componentes, un puente Ethernet a RS232 WIZ110SR de WIZnet y una tarjeta desarrollada a medida, basada a su vez en un microcontrolador AT89S52, que se encarga de retroalimentar el bootloader del MCU principal y de proporcionar los niveles lógicos demandados a las entradas del microcontrolador sobre el que se desarrolla la experimentación.

El dispositivo WIZ100SR implementa la Pila IP, soportando para el desarrollo de este experimento dos tipos de comandos HTTP: GET para indicar el valor de las entradas digitales desde el interfaz gráfico y POST para recibir el fichero binario que deberá ser volcado a través de interfaz de periféricos serie (SPI). Todos los comandos GET son interpretados por el MCU 89S52 que se encarga de volcar sobre las GPIOS apropiadas los valores demandados. Por medio de un comando POST, el estudiante envía desde el interfaz gráfico (Figura 2.23) el binario que contiene el desarrollo con el que desea experimentar, la tarjeta WIZ110SR envía dicho binario a través de su puerto RS232, que es recibido por el microcontrolador AT89S52 y volcado mediante SPI al bootloader del dispositivo análogo sobre el que se desarrolla la experimentación.

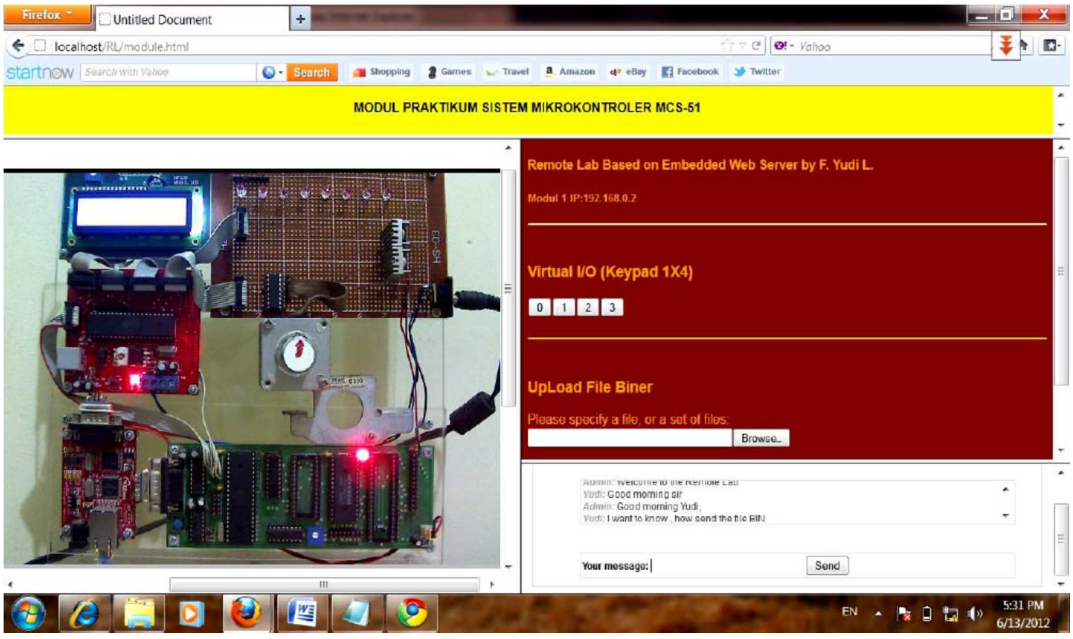

Figura 2.23. Interfaz gráfico de usuario del laboratorio remoto para experimentación con microcontroladores de la Universidad de Indonesia.

2.9.2.3 Plataforma de experimentación

En este laboratorio remoto la experimentación se desarrolla sobre un microcontrolador Atmel AT89S52. La programación de este microcontrolador se lleva a cabo mediante un bootloader instalado que altera los vectores de Reset e Interrupciones, condicionando las posiciones de la memoria de programa libres para ubicar el experimento del estudiante.

Respecto a los periféricos incluidos, este laboratorio permite la experimentación sobre una pantalla LCD alfanumérica, un grupo de 10 diodos led y un motor paso a paso. Además el usuario puede controlar desde el interfaz gráfico el valor de 4 entradas digitales.

2.9.2.4 Cliente

Como se ha comentado anteriormente, el estudiante desarrolla toda su experimentación sobre el laboratorio remoto desde una página web HTML que no requiere la instalación de complemento alguno.

Si bien la necesidad de generar un fichero ejecutable (binario) con unas restricciones particulares debido a la necesidad de adecuación del firmware desarrollado por el usuario con el bootloader desplegado en la plataforma de experimentación, exige el empleo de un entorno de desarrollo compatible con la programación de esta familia de microcontroladores (AT89S52) que permita alterar los vectores de Reset e Interrupciones. Los principales entornos que cumplen con esta característica están disponibles únicamente para plataformas Microsoft Windows (Atmel Studio) y muchos requieren de la adquisición de una licencia para permitir la adecuación exigida (Keil μVision, IAR).

2.9.3 Escenario de uso

La utilización de este laboratorio remoto es realmente sencillo. El estudiante debe conectarse al portal web principal, autenticarse y realizar una reserva sobre una de las instancias que se encuentre libre. A partir de ese momento la sesión de experimentación se establece y accede al interfaz gráfico de usuario (Figura 2.23) durante un tiempo máximo de 1 hora. Durante ese periodo el estudiante puede cargar el programa en la plataforma de experimentación y monitorizar el comportamiento del mismo a través de la webcam. Desde el interfaz gráfico de usuario el estudiante puede además alterar el valor de 4 entradas digitales.

El estudiante debe llevar a cabo todo el proceso de edición y compilación del programa independientemente del laboratorio remoto pero teniendo en cuenta las restricciones exigidas por el bootloader que permite la reprogramación en tiempo de ejecución del microcontrolador, requiriendo un conjunto de configuraciones que difieren de las que deberían de llevarse a cabo en un laboratorio presencial.

2.9.4 Resumen

Este laboratorio ha sido desarrollado teniendo en cuenta dos características fundamentales: permitir la escalabilidad (1) y minimizar el coste (2). Ambas características han sido conseguidas gracias a la utilización de un microservidor con conectividad Ethernet conectado independientemente al servidor central y que se encarga de la gestión integral de la plataforma de experimentación.

El desarrollo de este microservidor se ha llevado a cabo atendiendo a los requisitos específicos del laboratorio. Esto acaba con cualquier posibilidad de adecuar la solución hacia otras plataformas de experimentación. Por otro lado la elección de un dispositivo puente Ethernet-RS232 no resulta comprensible, atendiendo a la multitud de dispositivos programables que incorporan directamente conectividad Ethernet.

Respecto a la programación del microcontrolador, esta es llevada a cabo mediante el empleo de un bootloader grabado en el mismo a través de un interfaz de periféricos serie (SPI). Esta solución exige alterar el posicionamiento físico del inicio del programa (vector de reset), la dirección de las rutinas de gestión de interrupción e impide el acceso a ciertos registros especiales. Por tanto, se trata de un bootloader sumamente intrusivo que altera significativamente el desarrollo de los programas, exigiendo un conjunto de acciones específicas de este laboratorio remoto, con un alto condicionamiento en los entornos de desarrollo a emplear y la configuración de los mismos, que separa la experimentación efectuada de la llevada a cabo en un laboratorio presencial con el mismo dispositivo.

En cuanto a la usabilidad, el elevado tiempo de sesión definido para la experimentación, 1 hora, complica el acceso de grupos elevados de usuarios al laboratorio.

Finalmente, en lo referente al coste del laboratorio, el uso del bootloader abarata el coste final del laboratorio al evitar la adquisición de un dispositivo programador.

2.10 Laboratorio MicroLab de la Universidad Demirel Sulayman

Este laboratorio remoto (Kutlu & Taşdelen, 2010) (Kutlu, 2004) ha sido desarrollado por el departamento de educación en sistemas basados en computadores de la Facultad de Estudios Técnicos de la Universidad Demirel Sulayman en Turquía, dirigido por el Dr. Akif Kutlu.

Este laboratorio remoto es uno de los pioneros en tecnologías de sistemas embebidos puesto que la primera versión fue publicada en el año 2004 (Kutlu, 2004). Desde entonces el laboratorio ha sido mejorado y su uso ha sido continuado hasta el día de hoy.

2.10.1 Funcionalidad

Este laboratorio permite la experimentación con el microcontrolador 8051. Este longevo microcontrolador, cuyo lanzamiento data de 1980, sigue utilizándose ampliamente a día de hoy a consecuencia de las últimas mejoras efectuadas sobre la arquitectura original que lo convierten, por rendimiento y consumo, en una opción óptima para el desarrollo de aplicaciones embebidas que requieran un rendimiento limitado. El laboratorio remoto MicroLab permite la programación del microcontrolador y la interactuación en tiempo real mediante un conjunto de entradas y salidas, un panel LCD alfanumérico e incluso un brazo robótico de 5 ejes.

2.10.1.1 Integración con otras plataformas de aprendizaje o experimentación

Una de las particularidades de este laboratorio es que utiliza una aplicación de escritorio como cliente. Esto dificulta enormemente su integración con cualquier plataforma de aprendizaje colaborativo o sistemas de gestión de laboratorios remotos.

2.10.1.2 Escalabilidad

La característica fundamental de este laboratorio es la comunicación mediante bus CAN (Controller Area Network) entre el servidor y las plataformas de experimentación e interacción. El número de nodos conectados al bus CAN depende de los parámetros configurados en los transceptores, utilizando 11 bits (2048 nodos) en la implementación del laboratorio remoto MicroLAB. Dado que cada experimento consume dos nodos, la escalabilidad queda garantizada. Actualmente el laboratorio Microlab dispone de tres instancias desplegadas (Figura 2.24) que permiten respectivamente la experimentación con entradas y salidas digitales (1), un panel LCD alfanumérico (2) y un brazo robótico (3).

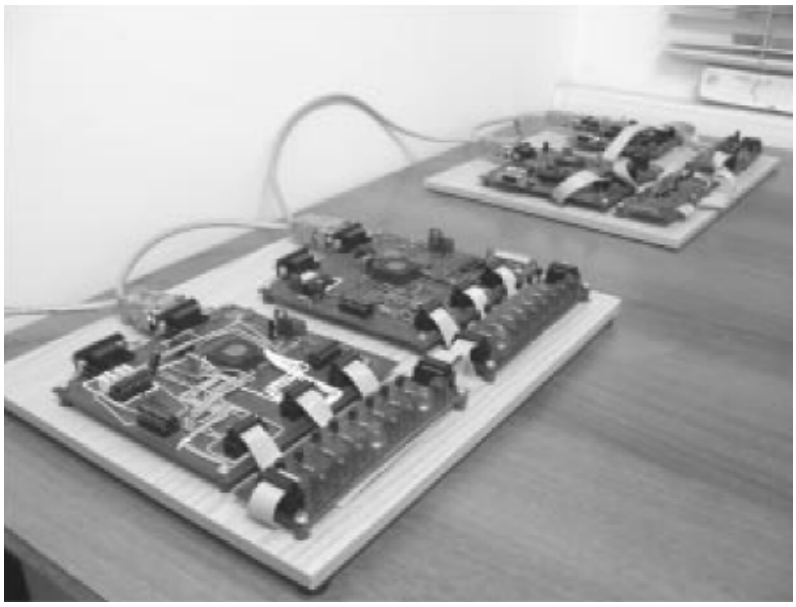

Figura 2.24. Dos instancias de experimentación del laboratorio remoto MicroLab.

2.10.1.3 Desplegabilidad

Todo el software y hardware desarrollados para el despliegue del laboratorio remoto pertenece a los desarrolladores del mismo.

Además, la autenticación de los usuarios requiere de un servidor Microsoft SQL Server para alojar la información de los usuarios con acceso y el registro de las sesiones efectuadas.

2.10.1.4 Replicabilidad

La arquitectura de este laboratorio complica la adecuación del laboratorio hacia otra tecnología. La programación del microcontrolador que soporta la experimentación se lleva a cabo mediante un bootloader a través de un canal UART. Esta metodología facilita la reprogramación desde el bus CAN, pero limita las plataformas de experimentación a microcontroladores con capacidad para alterar en tiempo de ejecución su memoria de programa.

2.10.1.5 Disponibilidad

Como se ha comentado anteriormente, el equipo desarrollador lleva utilizando este laboratorio durante más de 10 años en las asignaturas básicas de diseño de sistemas basados en microprocesadores de la Facultad de Estudios Técnicos de la Universidad Demirel Sulayman.

Aunque el sistema dispone de un sistema propio de autenticación, es posible acceder al laboratorio como invitado. A través de la página http://microlab.sdu.edu.tr/ se puede descargar el programa cliente (microclient) con el que accedemos al laboratorio.

Actualmente sólo dos experimentos están accesibles (Figura 2.24) que permiten experimentar con los puertos digitales del microcontrolador 8051 y con una pantalla LCD alfanumérica.

2.10.1.6 Conectividad

Toda la comunicación entre el servidor (microserver) y el cliente (microclient) se lleva a cabo mediante protocolo de acceso a objetos simple (SOAP). El servidor dispone de múltiples sockets listener que permiten establecer comunicación con múltiples clientes a través de los puertos TCP entre el 2000 y el 2020. Esto exige la apertura de dichos puertos y complica la conectividad en instalaciones con conexión a Internet detrás de un firewall.

2.10.1.7 Universalidad

La única forma de utilizar el laboratorio es utilizando la aplicación de escritorio proporcionada por los desarrolladores. Esta aplicación está desarrollada mediante Visual C# y se encuentra disponible únicamente para plataformas Windows.

Además es necesario un compilador para el microcontrolador 8051 que genere archivos binarios en formato Intel hex. Al ser un dispositivo tan ampliamente extendido existen múltiples compiladores que cumplen esa condición, tanto para dispositivos con sistema operativo Microsoft Windows como Linux, sin embargo esta característica impide el desarrollo de todas las etapas de la experimentación con este laboratorio desde dispositivos móviles (Android, iOS, etc…)

2.10.2 Implementación

Como se ha indicado anteriormente la característica más reseñable de este laboratorio es la conexión entre el servidor principal y las instancias de experimentación a través del bus CAN.

La Figura 2.25 muestra la arquitectura de este laboratorio remoto.

2.10.2.1 Servidor principal

Al igual que en los dispositivos clientes, el servidor principal debe ejecutar una aplicación desarrollada a medida para soportar las comunicaciones del laboratorio en Visual Studio. Esto exige la adopción de un servidor basado en el sistema operativo Microsoft Windows. Básicamente la aplicación MicroServer mantiene un socket TCP listener en el servidor que permite el establecimiento de una conexión con la aplicación cliente. A través de este socket, mediante protocolo SOAP, el cliente puede enviar los ficheros subidos por los estudiantes así como cualquier cambio en las entradas digitales que deba ser enviada a la plataforma de experimentación, de la misma forma que el servidor envía periódicamente el valor de las salidas. Para minimizar las comunicaciones, únicamente se transmite información cuando el sistema detecta un cambio.

Inicialmente el servidor permite la comunicación en el puerto 2000, generando otro socket listener en el siguiente puerto cuando éste es consumido por un cliente. El número máximo de conexiones simultaneas es de 21 (puertos 2000 al 2020). La aplicación cliente Microclient comienza probando por el primero hasta que encuentra uno libre. En el caso de estar todos ocupados, devuelve un error emplazando al estudiante a probar transcurridos unos minutos.

La comunicación del servidor con las instancias de experimentación se desarrolla mediante bus CAN mediante una tarjeta PCIcanD de Agilent que debe ser instalada en el servidor. La propia aplicación Microserver dispone de un módulo capaz de enviar el

programa del estudiante, así como los valores de las entradas durante la experimentación a la instancia correspondiente mediante el bus CAN.

El laboratorio dispone de un sistema de autenticación y registro de sesiones que utiliza un sistema de gestión de base de datos Microsoft SQL Server. Además de almacenar los datos de los usuarios registrados se almacenan los inicios de sesión y todas las conversaciones mantenidas mediante un sistema de chat entre los distintos usuarios del laboratorio.

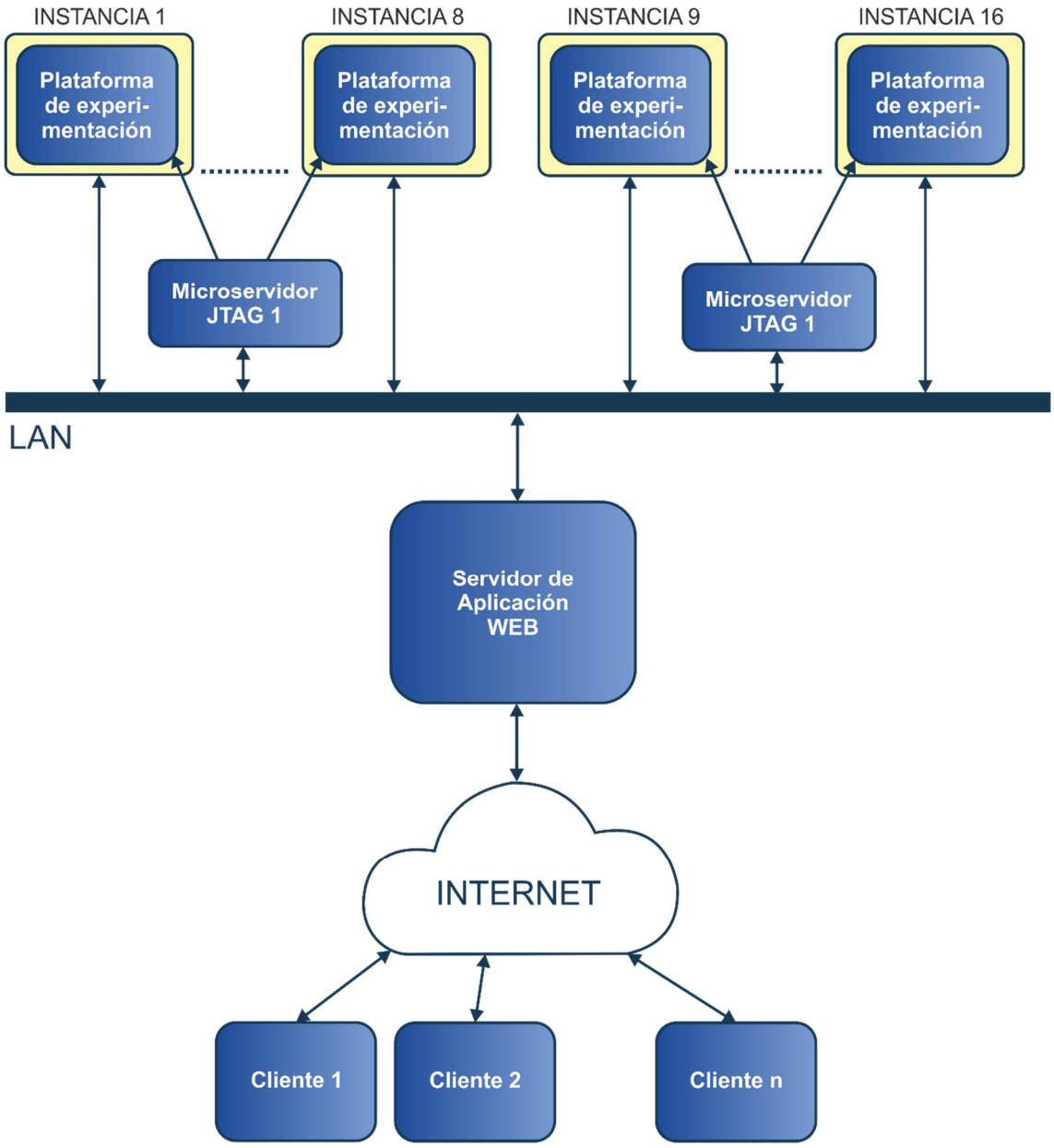

Figura 2.25. Arquitectura del laboratorio remoto MicroLab desarrollador en la la Facultad de Estudios Técnicos de la Universidad Demirel Sulayman.

2.10.2.2 Plataforma de experimentación

La plataforma de experimentación está basada en una tarjeta de desarrollo sobre un microcontrolador 8051 conectado al bus CAN mediante un transceptor 89C51CC01 de Intel a través del puerto UART. El microcontrolador lleva programado un programa bootloader que permite reprogramar su memoria de programa con el programa enviado desde el servidor mediante el bus CAN.

El firmware de cada instancia debe contener el ID del nodo para permitir discriminar los programas que son enviados hacia otras instancias.

La tarjeta de desarrollo dispone de conexiones con los periféricos de salida de cada experimento a través del puerto P1 y con las entradas enviadas desde la tarjeta de interacción mediante el puerto P2.

2.10.2.3 Tarjeta de interacción

La tarjeta de interacción está implementada sobre el mismo sistema de desarrollo sobre el cual se ejecuta la experimentación. En este caso el microcontrolador 8051 dispone de un firmware que recibe mediante el bus CAN los comandos enviados por los estudiantes que permiten la alteración de las entradas, alterando en consecuencia los pines de salida correspondientes.

Al igual que ocurría con la plataforma de experimentación, el firmware programado en el microcontrolador 8051 de cada instancia debe contener un identificador (CAN ID) único que permita discriminar los paquetes de los que es el destinatario.

2.10.2.4 Cliente

Como se ha indicado a lo largo de este apartado, la única forma de acceder a la experimentación con este laboratorio es mediante la aplicación Microclient. Esta aplicación ha sido desarrollada con Microsoft Visual Studio y únicamente es compatible con plataformas con sistema operativo Microsoft Windows (Figura 2.26).

Esta aplicación establece un socket con el servidor y muestra un interfaz gráfico de usuario que permite la experimentación. Toda la comunicación entre cliente y servidor se lleva a cabo mediante protocolo SOAP.

El estudiante puede experimentar con el laboratorio programando la plataforma de experimentación con un fichero binario en formato Intel hex que debe ser generado por el usuario utilizando una de los múltiples compiladores o entornos de desarrollo disponibles para el microcontrolador 8051. La aplicación cliente verifica el correcto formato del archivo binario y altera las posiciones de memoria para adecuar el código generado por el estudiante al bootloader que se encarga de reprogramar la memoria de programa del microcontrolador. Además este bootloader no utiliza interrupciones, de

modo que no afecta a la gestión que el estudiante hace de las mismas. De esta forma el programa desarrollado por el estudiante es idéntico a aquel que implementaría en un laboratorio presencial.

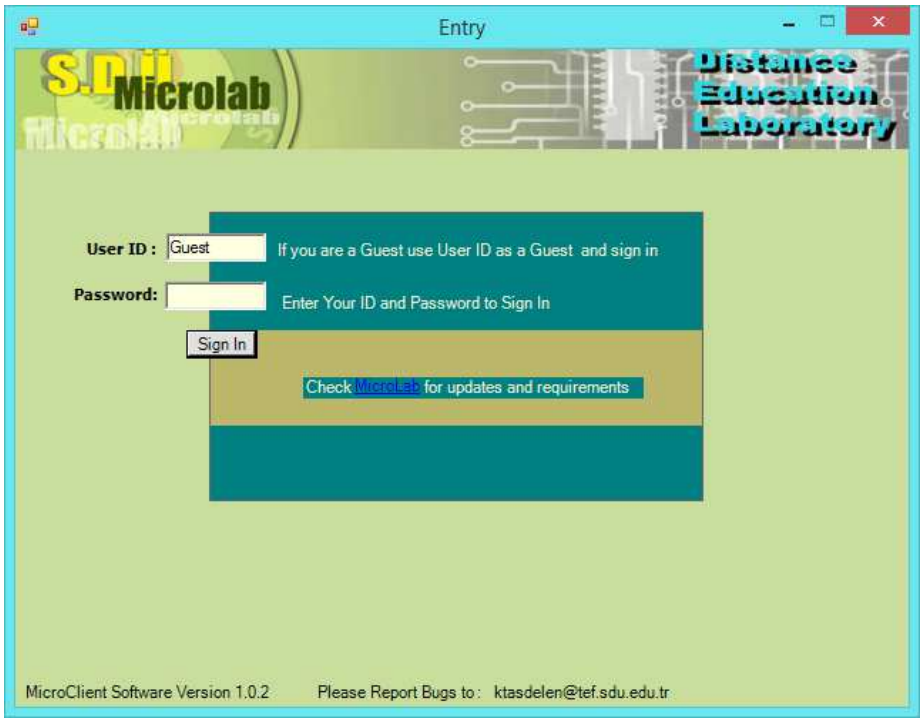

Figura 2.26. Captura de la aplicación cliente del laboratorio Microlab para experimentación con el microcontrolador 8051.

2.10.3 Escenario de uso

Una vez el estudiante se identifica, la aplicación muestra una ventana denominada Lobby, en la que se listan los experimentos disponibles dentro del laboratorio remoto indicando el estado de cada uno de ellos. Cada experimento puede estar libre u ocupado. Si el experimento se encuentra libre, el estudiante puede acceder en modo supervisor. Este modo le permite reprogramar el microcontrolador e interactuar desde las entradas que se visualizan en el interfaz gráfico de usuario. En caso de que el experimento se encuentre ocupado, el estudiante puede acceder en modo pasivo, pudiendo observar la experimentación llevada a cabo por el supervisor. Entre todos los usuarios que han accedido a un experimento se establece un chat.

El tiempo dedicado para supervisar un experimento por cada estudiante es de 10 minutos transcurridos los cuales el control pasa al siguiente estudiante que ha accedido como usuario pasivo.

2.10.4 Resumen

Este laboratorio ha sido diseñado para permitir el control directo de múltiples instancias de experimentación desde el servidor principal. La principal característica es que la comunicación entre el servidor y las instancias se lleva a cabo mediante bus CAN. La adopción de este protocolo garantiza la escalabilidad del laboratorio pero paralelamente exige la instalación de hardware compatible en el servidor e impide la virtualización del mismo. Además el bus CAN exige una infraestructura de comunicaciones propia entre todos los componentes del laboratorio y proporciona una velocidad de transmisión máxima de 1Mbps. Por otro lado la programación de la plataforma de experimentación desde el propio bus CAN exige el desarrollo de un bootloader específico y limita la adecuación del laboratorio a tecnologías que sean reprogramables mediante esta técnica. Dicho de otra forma el bus CAN no aporta ninguna ventaja sobre Ethernet y añade múltiples limitaciones.

En otro sentido otra de las particularidades del laboratorio se centra en la exigencia de empleo de una aplicación de escritorio como interfaz gráfico de usuario. Aunque la estabilidad y simplicidad de la misma sean aceptables, esta decisión exige que el acceso del estudiante se produzca desde un computador con sistema operativo Microsoft Windows y con permisos para la instalación de nuevo software.

2.11 Arduino Remote lab en la Universidad Abierta Helénica (HOU),

Este laboratorio remoto ha sido desarrollado en la Universidad Abierta Helénica (HOU) bajo la supervisión del Dr. Vassilis Fotopoulos. Este laboratorio remoto (Fotopoulos, Anastasios, & Anastasios, 2013) está desplegado en las instalaciones del laboratorio de Sistemas y Medios de Comunicación Digital Informática (DSMC) del HOU, dirigido por el Prof. Athanassios Skodras, y soportado por la Sociedad de Sistemas y Circuitos de IEEE (IEEE CAS).

2.11.1 Funcionalidad

Este proyecto está diseñado para proporcionar un laboratorio remoto sobre el que realizar una serie de ejercicios basados en la plataforma Arduino. Todo el proceso de experimentación incluyendo la edición del programa, grabación del microcontrolador y monitorización del comportamiento puede ser llevado a cabo dentro del laboratorio remoto.

2.11.1.1 Integración con otras plataformas de aprendizaje o experimentación

No se ha contemplado su integración en ninguna plataforma de aprendizaje colaborativo ni en sistemas de gestión de laboratorios remotos.

2.11.1.2 Escalabilidad

La escalabilidad no ha sido contemplada durante el desarrollo del sistema. Desde el punto de vista hardware, la conexión de la plataforma de experimentación con el servidor se realiza mediante USB utilizando un puente USB/RS232 integrado en la plataforma de explotación. Este sistema permite la conexión de varias plataformas de experimentación idénticas sobre un mismo servidor utilizando distintos puertos COM. Si bien el software del servidor ha sido desarrollado sin tener en cuenta esta posibilidad por lo que sería necesario adaptar el sistema de reservas y la aplicación web para soportar múltiples instancias.

2.11.1.3 Desplegabilidad

Todo el diseño de este laboratorio se ha llevado a cabo teniendo en cuenta como principales características su bajo coste y facilidad para desplegarlo en otras instalaciones. Todos los componentes software del servidor han sido desarrollados mediante tecnologías de software libre. Así mismo la plataforma hardware ha sido diseñada utilizando componentes fácilmente adquiribles. La conexión de los diferentes periféricos con la plataforma de experimentación se ha llevado a cabo mediante tarjetas de prototipo rápido, de forma análoga a la que los estudiantes implementan los prototipos en laboratorios presenciales (Figura 2.27). Esta decisión reduce la integridad del laboratorio pero facilita su despliegue en otras instalaciones.

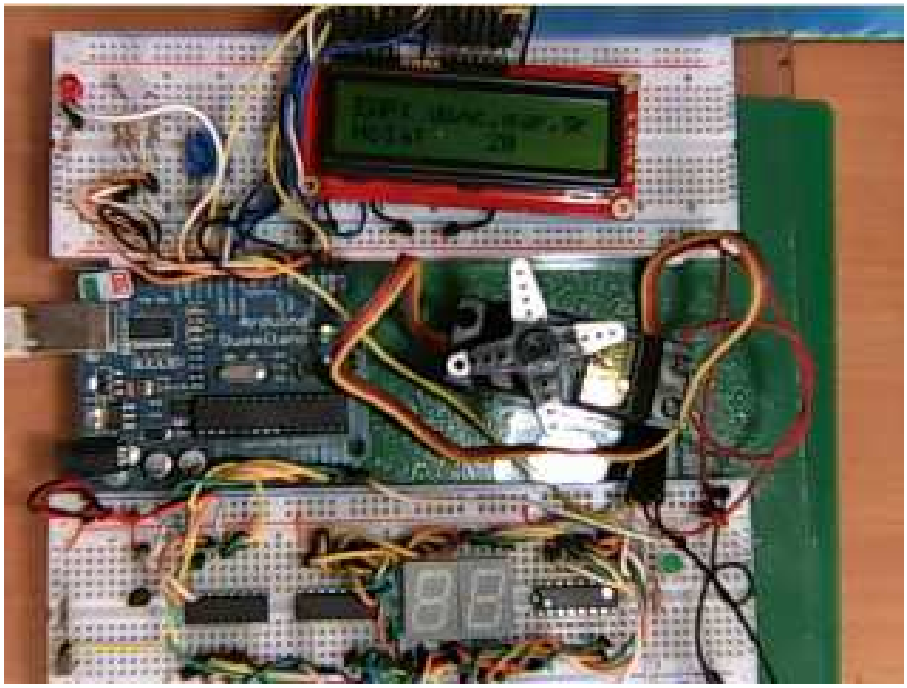

Figura 2.27. Plataforma de experimentación del Arduino Remote Lab en la Universidad Abierta Helénica (HOU).

2.11.1.4 Replicabilidad

El laboratorio remoto ha sido diseñado específicamente atendiendo a los requisitos del experimento soportado. Únicamente es adaptable a otras tarjetas basadas en la plataforma de hardware libre Arduino. Tanto el compilador integrado en el cliente web como el sistema de programación son exclusivos de esta plataforma, siendo imposible replicar este laboratorio hacia otras tecnologías o familias de microcontroladores.

2.11.1.5 Disponibilidad

Este laboratorio remoto está financiado por la Sociedad de Sistemas y Circuitos de IEEE (IEEE CAS). Aunque es necesario autenticarse, el laboratorio permite el registro de nuevos usuarios sin requerir ningún tipo de validación.

El laboratorio proporciona información sobre los periféricos conectados y el diagrama de conexionado. Además el estudiante dispone de 7 ejercicios comentados que puede modificar directamente desde el editor online que aparece en el interfaz de usuario (Figura 2.28).

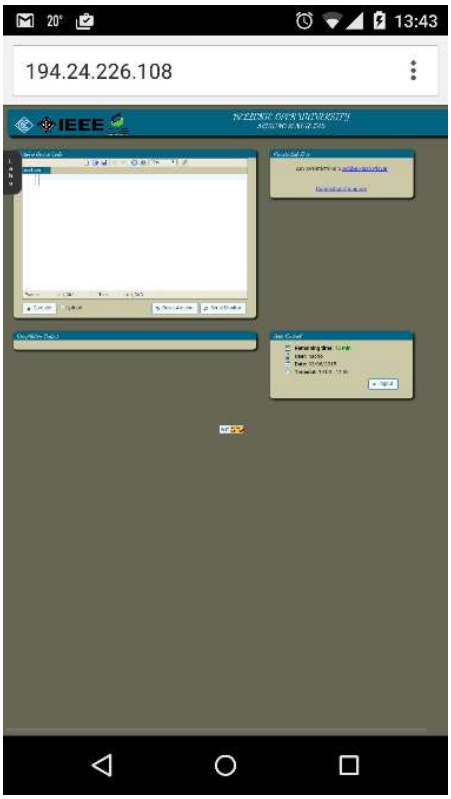

Figura 2.28. Acceso al laboratorio remoto para experimentación con Arduino de la Universida Abierta Helénica (HOU) desde un smartphone Android.

2.11.1.6 Conectividad

Este laboratorio proporciona todos los recursos necesarios para experimentar con la plataforma Arduino Duemilanove. A través del navegador el estudiante puede

autenticarse, reservar el laboratorio en intervalos de una hora y llevar a cabo la experimentación. Ninguna configuración particular de la infraestructura de red es necesaria para acceder a este laboratorio.

2.11.1.7 Universalidad

Pese a que toda la experimentación se lleva a cabo desde un navegador, el funcionamiento del cliente web exige la ejecución de applets de Java y Flash, limitando el acceso desde dispositivos móviles. Además la web no es responsiva a diferentes resoluciones de pantalla siendo poco natural el acceso al laboratorio desde dispositivos móviles como puede observarse en la Figura 2.28.

2.11.2 Implementación

El esquema del laboratorio se puede observar en la Figura 2.29. Es una arquitectura muy básica formada por un servidor principal que se encarga de realizar todas las tareas de administración y la programación de la plataforma de experimentación.

2.11.2.1 Servidor principal

El servidor principal implementa un sistema Linux sobre el que se ha instalado XAMPP con una versión actualizada de Apache y MySQL. La monitorización del experimento se lleva a cabo mediante una webcam de bajo coste utilizando el encoder de video nativo de Linux (avconv/ffmpeg) y un servidor de streaming abierto.

La programación de la plataforma de experimentación se lleva a cabo mediante el propio entorno de desarrollo integrado proporcionado por la plataforma Arduino para Linux.

Para el desarrollo del laboratorio se ha desarrollado un sistema de reserva que permite a los estudiantes registrados establecer franjas de acceso de una hora. Toda la información relativa a los usuarios y reservas se almacena en la base de datos sobre el SGBD MySQL.

Una de las características de este laboratorio remoto es que al no permitir el acceso concurrente al mismo deriva en un rendimiento limitado. Para la implementación del servidor principal se ha utilizado un ordenador escritorio basado en un procesador Pentium IV con 512 MB de memoria RAM y únicamente 40GB de espacio en disco.

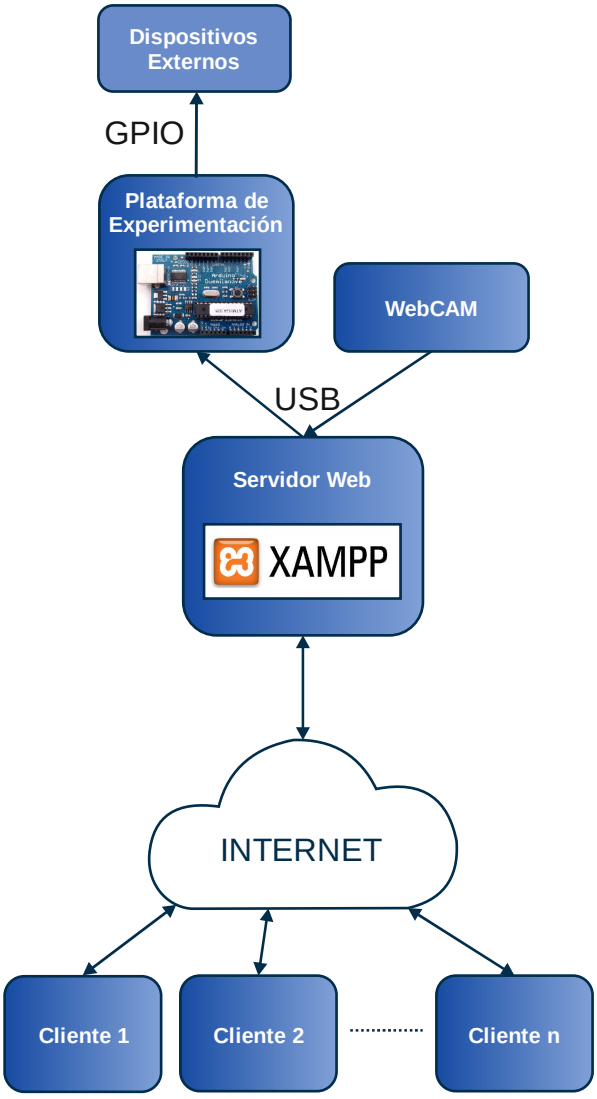

Figura 2.29. Arquitectura del laboratorio remoto para experimentación con Arduino de la Universidad Abierta Helénica (HOU).

2.11.2.2 Cliente

El interfaz de usuario con el laboratorio se desarrolla mediante un navegador. El cliente utiliza tecnología AJAX, applets de Java para soportar el editor de código y el terminal serie utilizado para el debugging, y Flash para la monitorización de la cámara. Una limitación importante de este laboratorio es que carece de capacidad de interacción con la plataforma de experimentación. El estudiante puede programar el microcontrolador y monitorizar su funcionamiento pero no interactuar con el experimento, careciendo de un panel de control de las entradas en el interfaz gráfico de usuario. La captura del experimento puede verse en la Figura 2.30.

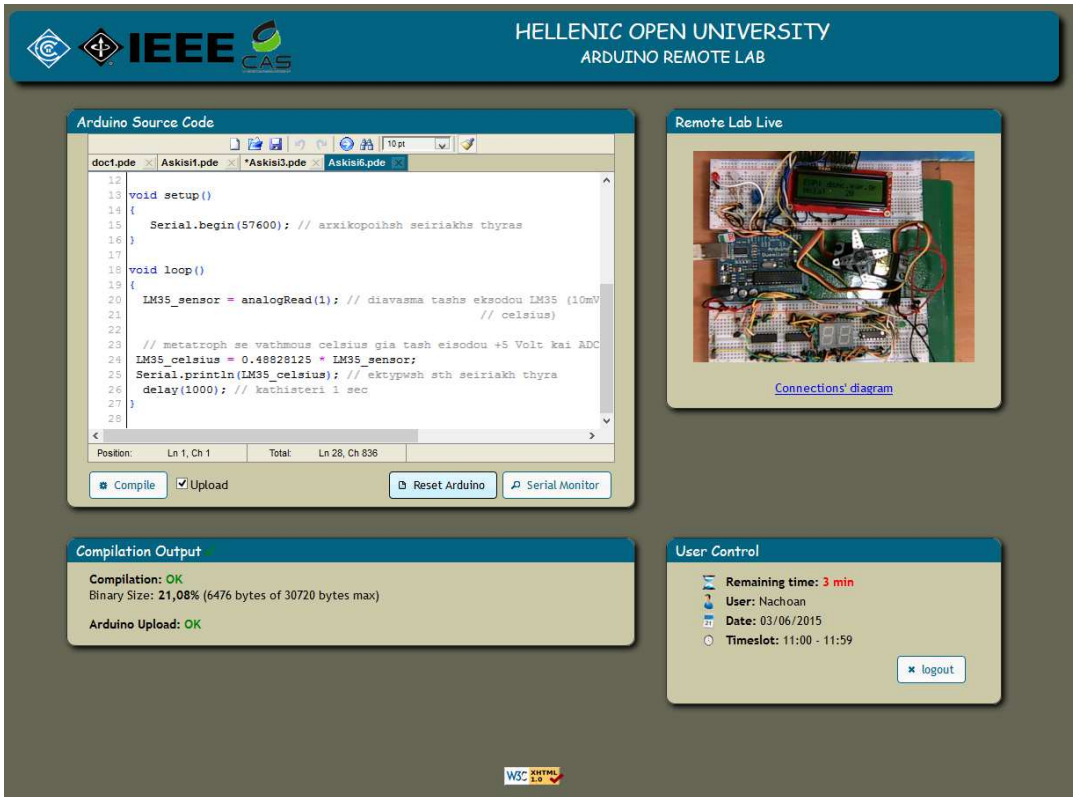

Figura 2.30. Interfaz de usuario del laboratorio remoto para experimentación con Arduino de la Universidad Abierta Helénica (HOU).

2.11.2.3 Plataforma de experimentación

La plataforma de experimentación está formada por una tarjeta Arduino Duemilanove y un conjunto de periféricos:

- Sensor de temperatura LM35
- LCD alfanumérico de 2 líneas x 16 caracteres
- Servomotor MG995
- Diodo led asociado a salida PWM
- Diodo led RGB
- Dos displays de 7 segmentos

El diagrama de conexionado se puede observar en la Figura 2.31.

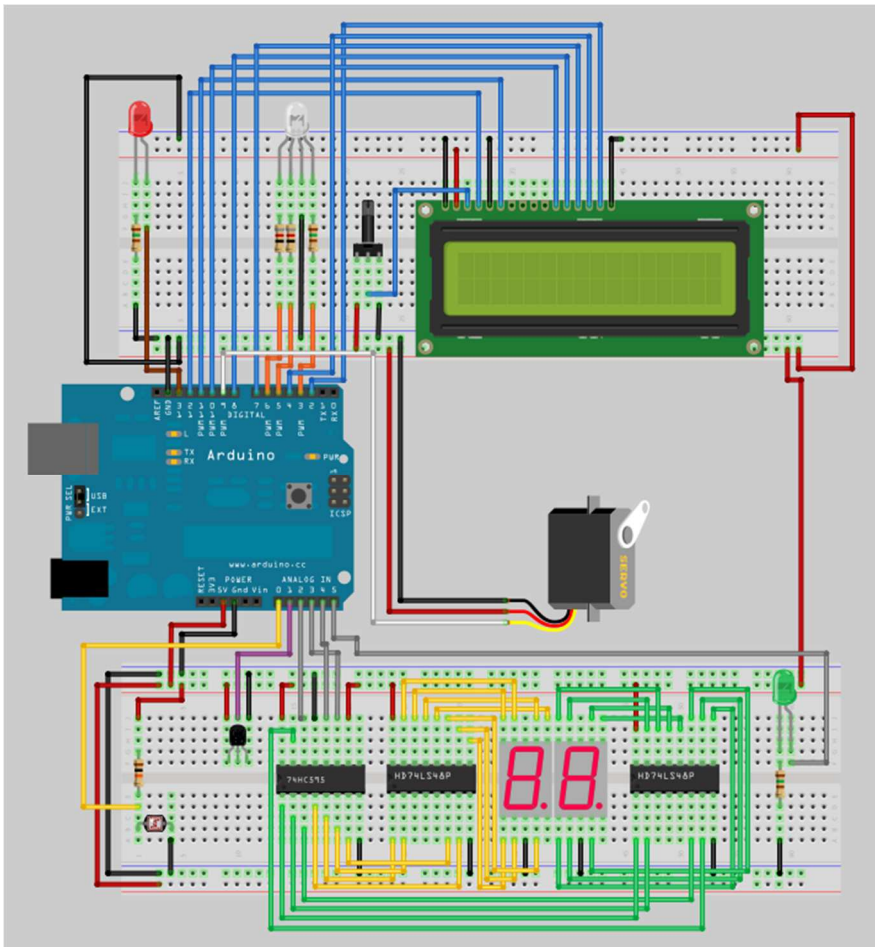

Figura 2.31. Diagrama de conexionado del laboratorio remoto para experimentación con Arduino de la Universidad Abierta Helénica (HOU).

2.11.3 Escenario de uso

Para llevar a cabo la experimentación en este laboratorio remoto, el estudiante debe previamente realizar una reserva. Cada reserva tiene una duración de una hora completa del día y un estudiante no puede mantener más de tres reservas simultáneas. El cliente web dispone de una herramienta llevar a cabo una reserva, así como para modificar o borrar las reservas previamente efectuadas.

Si durante una reserva propia el estudiante se registra, puede acceder al laboratorio a través de un interfaz gráfico de usuario que se muestra en la Figura 2.30. Desde este GUI el usuario puede editar el programa utilizando un editor asistido integrado en el mismo, llevar a cabo la compilación y programación del MCU y monitorizar el comportamiento desde la webcam. El interfaz de usuario dispone de un monitor para el puerto serie del Arduino que proporciona una herramienta básica de depuración para comprobar la ejecución del programa.

Un menú lateral permite cargar en el editor 7 ejercicios básicos que documentan el funcionamiento de los diferentes periféricos (Figura 2.32).

Durante el tiempo reservado el estudiante puede llevar a cabo tantas programaciones del experimento como desee. Diez minutos antes de finalizar el tiempo de la reserva el cliente muestra una venta emergente que muestra el tiempo de sesión restante. Cuando la sesión finaliza, el sistema retorna automáticamente a la web de reservas.

2.11.4 Resumen

La característica fundamental de este laboratorio es sin duda el bajo coste requerido para su despliegue. El total de los materiales utilizados no supera los 100€, descontando el coste del servidor, que como se ha indicado puede implementarse en cualquier computador de escritorio descatalogado.

Además la razón que ha motivado la inclusión de este laboratorio en este análisis es la inclusión del entorno de desarrollo requerido para la experimentación en el propio cliente. Esto permite desarrollar las tareas de codificación y grabación desde la propia sesión en el laboratorio.

Sin embargo, una limitación importante es la carencia de interacción con el experimento al no permitir alterar en tiempo real señales de entrada conectadas al microcontrolador sobe el que se desarrolla la experimentación.

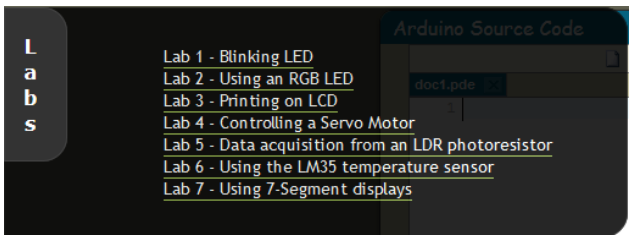

Figura 2.32. Catálogo de ejercicios disponibles en el laboratorio remoto para experimentación con Arduino de la Universidad Abierta Helénica (HOU).

2.12 Laboratorio remoto de la Universidad de Maribor para experimentación con DSP

Este laboratorio ha sido desarrollado por el Instituto de Robótica de la Universidad de Maribor, bajo la supervisión del Dr. Darko Hercog (Hercog, Gergic, Uran, & Jezernik, 2007) (Rojko, Hercog, & Jezernik, 2010) (Uran, Hercog, & Jezernik, 2006) (Hercog & Jezernik, 2005). Fue desarrollado en el año 2007 para permitir a los estudiantes realizar experimentación remota en control automático.

2.12.1 Funcionalidad

Este laboratorio remoto es la evolución natural de una tarjeta de desarrollo educacional para el prototipado rápido de sistemas de control implementada por el Instituto de

Robótica (FERI) de la Facultad de Ingeniería Eléctrica y Ciencias de la Computación de la Universidad de Maribor. Este sistema de desarrollo, denominado DSP-2 (Figura 2.33), ha sido utilizado con éxito en cursos de sistemas de control impartidos desde el año 2005 (Jezernik, 2005). Implementado en base a un procesador en coma flotante TMS320C32 de Texas Instruments, su principal funcionalidad es permitir implementar mediante MATLAB/Simulink sistemas de control, que pueden ser validados utilizando LabView. Una de las partes fundamentales es el desarrollo de una librería para Simulink que incorpora en la "toolbox" un conjunto de bloques con controladores para todos los puertos de entradas y salidas del sistema de desarrollo. De esta forma el estudiante puede desarrollar algoritmos de control utilizando la herramienta de diseño rápido basada en bloques de Simulink con total abstracción de la programación del DSP y la configuración de los equipos de instrumentación.

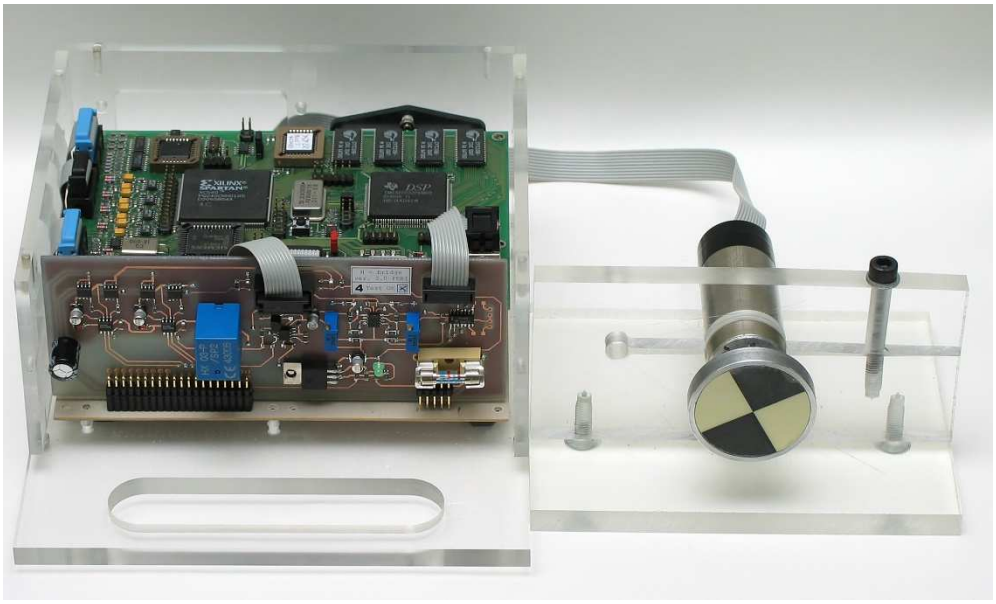

Figura 2.33. Imagen del módulo de aprendizaje DSP-2 conectado a un motor de continua.

Los buenos resultados obtenidos por los estudiantes en los laboratorios presenciales motivaron al equipo desarrollador a lanzar un laboratorio remoto que permitiera a los estudiantes desarrollar esa experimentación remotamente.

2.12.1.1 Integración con otras plataformas de aprendizaje o experimentación

Este laboratorio remoto está compuesto por tres componentes: el sistema de desarrollo DSP-2, el servidor de laboratorio que permite experimentar mediante los paneles de control generados con LabView y un servidor de administración que se encarga de la autenticación y el sistema de reserva. Aunque el servidor de administración puede implementarse en el mismo equipo donde se aloja el servidor del laboratorio, en la implementación se han separado debido a que los servicios de administración han sido integrados en el sistema de gestión del aprendizaje de la Facultad de Ingeniería Eléctrica y Ciencias de la Computación de la Universidad de Maribor, basado en la plataforma Moodle.

2.12.1.2 Escalabilidad

Esa independencia entre el servidor de administración y el servidor del laboratorio permite incorporar múltiples instancias al laboratorio. Para cada instancia se requiere un servidor de laboratorio independiente que se encarga tanto de llevar a cabo la programación del DSP mediante puerto serie como de monitorizar la señal en tiempo real y permitir el ajuste de parámetros. Cada instancia de experimentación puede ser integrada con independencia en el sistema de gestión del aprendizaje.

2.12.1.3 Desplegabilidad

No es posible desplegar este laboratorio en otras instalaciones porque tanto el sistema de desarrollo Hardware DSP-2, como la librería de Simulink que permite desarrollar aplicaciones para la misma, son propiedad del equipo desarrollador.

Además requiere de licencias de Matlab/Simulink y LabView en los servidores de laboratorio que encarecen enormemente el despliegue del laboratorio remoto.

2.12.1.4 Replicabilidad

La experimentación con este laboratorio está fundamentada en la librería de Simulink para el sistema de desarrollo DSP-2, por tanto es imposible su adecuación a otras tecnologías. La adecuación de este experimento a otro dispositivo DSP es complicada debido a las enormes diferencias entre las diversas arquitecturas de procesadores digitales de señal que exigirían un desarrollo completamente nuevo de la librería modular para Simulink.

2.12.1.5 Disponibilidad

El curso está disponible para los alumnos de la Facultad de Ingeniería Eléctrica y Ciencias de la Computación de la Universidad de Maribor a través del sistema de gestión de contenidos (CMS) corporativo basado en Moodle. Todos los experimentos del curso de prototipado rápido de sistemas de control se encuentran documentados. Actualmente el laboratorio permite la experimentación con los siguientes elementos:

- Oscilador RC;
- Velocidad de un motor de continua;
- Control en cascada de un motor de continua;
- Telecontrol de sistemas mecatrónicos

2.12.1.6 Conectividad

Este laboratorio no requiere ninguna configuración de la infraestructura de red específica para su funcionamiento. Toda la comunicación del estudiante con el laboratorio se lleva a cabo a través de un servidor web sobre el puerto 80.

2.12.1.7 Universalidad

Pese a que la experimentación se lleva cabo desde un navegador, es necesario disponer del plugin de LabView para poder acceder a los paneles de control de los experimentos generados mediante LabView Virtual Instruments y publicados mediante la herramienta LabView web Publishing. Esto limita la capacidad de acceder desde dispositivos móviles, pero facilita el acceso a instrumentos virtuales ya implementados en LabView que son fundamentales para la experimentación con procesadores digitales de señal.

Además la implementación de los algoritmos de control que son programados en el sistema de desarrollo DSP-2 deben desarrollarse mediante la herramienta MATLAB/Simulink. Esto limita los dispositivos clientes a ordenadores de escritorio basados en sistemas operativos Microsoft Windows o Linux y con unos requisitos avanzados de rendimiento.

2.12.2 Implementación

La arquitectura del laboratorio se puede observar en la Figura 2.34.

2.12.2.1 Servidor de administración

Este componente lleva a cabo las tareas de autenticación y reserva del laboratorio. Ambas tareas se han implementado como módulos integrables en el sistema de gestión de aprendizaje Moodle.

Este laboratorio utiliza el sistema de autenticación nativo de Moodle e incorpora un sistema de reservas desarrollado como extensión de Moodle y basado en el sistema implementado dentro del proyecto MARVEL (Ferreira & Cardoso, 2005), financiado por el programa Leonardo Da Vinci de la Comunidad Europea. Cada experimento contiene asociado una página web donde por medio de un calendario los estudiantes pueden reservar cada instancia en intervalos de una hora. Cuando los estudiantes acceden a la plataforma Moodle dentro del intervalo reservado el calendario incluye un enlace que permite acceder al experimento.

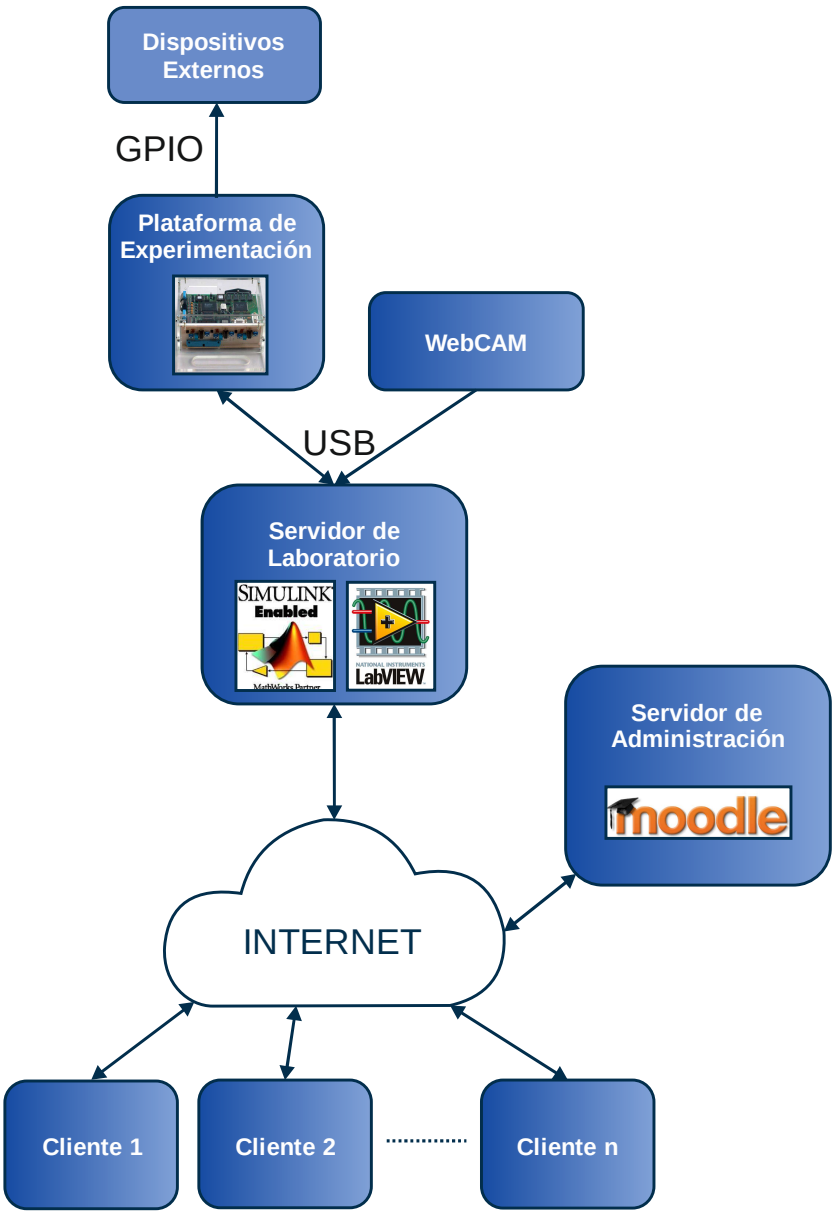

Figura 2.34. Arquitectura del laboratorio remoto basado en DSP de la Universidad de Maribor.

2.12.2.2 Servidor de laboratorio

Requiere la ejecución de los diferentes instrumentos virtuales (VI) implementados en LabView para monitorizar cada entrada o salida vinculada a un experimento, así como el servidor de LabView con el fin de permitir la experimentación remota. Cada VI individual permite intercambiar datos entre el sistema de control DSP-2 y el servidor de laboratorio, mientras que el servidor de LabView permite la publicación web de cada VI a través de la herramienta LabView web Publishing. Mediante esta herramienta los cuadros de control implementados en LabView se pueden publicar de forma rápida y sencilla en la web sin necesidad de programación adicional. Una vez que el VI se publica, cualquier usuario de la web, con el permiso adecuado, puede acceder y controlar un experimento utilizando un navegador web estándar con el correspondiente plugin de LabView instalado.

Después, cuando el estudiante accede al experimento, los controles de la interfaz gráfica de usuario se activan y ejecutan la aplicación de LabView como se desarrollaría en un ordenador local (Figura 2.35). Durante la ejecución del experimento remoto, el estudiante puede ajustar los parámetros del controlador y observar los valores de salida de proceso en texto o en modo gráfico. Nativamente los cuadros de control implementados mediante LabView incorporan un algoritmo que impide que más de un usuario pueda tomar el control sobre el experimento a distancia.

Otra de las tareas del servidor de laboratorio es proporcionar al estudiante las imágenes obtenidas desde la webcam. Nuevamente esta funcionalidad se ha implementado mediante un instrumento virtual de LabView. El VI captura imágenes directamente desde la webcam y las transmite comprimidas en formato jpeg.

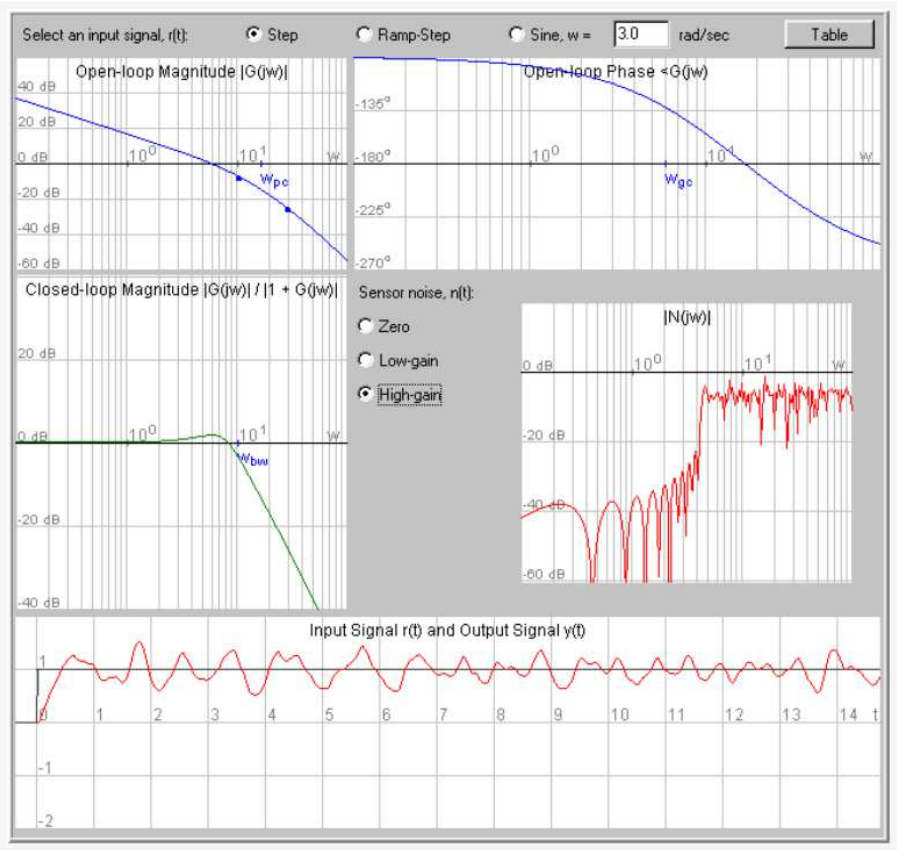

Figura 2.35. Cuadro de control de la instancia del laboratorio remoto basado en DSP de la Universidad de Maribor que permite la experimentación con servomotores.

2.12.2.3 Plataforma de experimentación

El componente fundamental de la plataforma de experimentación es el controlador DSP-2 que fue desarrollado en FERI (Figura 2.33). Los componentes clave del controlador DSP-2 son el procesador de coma flotante TMS320C32 fabricado por Texas Instruments (TI), que se utiliza para la ejecución del algoritmo de control; la FPGA integrada de la gama de la familia Xilinx Spartan 3, que implementa el modulador de ancho de pulso (PWM) y los interfaces para la interconexión de periféricos. Además, el controlador

DSP-2 contiene todos los periféricos necesarios, para el control de motores de corriente continua o alterna, convertidores Analógico-Digital (ADC) y Digital-Analógico (DAC), un PWM trifásico, un conjunto de entradas y salidas digitales de propósito general (GPIO) opto-acopladas, un interfaz para el encoder incremental, memoria RAM, memoria FLASH y un controlador de bus CAN.

2.12.2.4 Cliente

Para acceder al laboratorio es necesario disponer de un ordenador de escritorio con plataforma Microsoft Windows o Linux que permita la ejecución de la herramienta MATLAB/Simulink. Esta herramienta dispone de licencia comercial lo cual dificulta el acceso a la experimentación por parte de los alumnos.

Además una vez desarrollado el algoritmo de control la interacción con el laboratorio remoto se lleva a cabo mediante un navegador web que debe disponer del plugin de LabView.

2.12.3 Escenario de uso

Desde el sistema de gestión del aprendizaje de la Facultad de Ingeniería Eléctrica y Ciencias de la Computación de la Universidad de Maribor, el estudiante puede descargar la librería para la herramienta Simulink desarrollado para la gestión del entorno de desarrollo DSP-2. Esta librería incorpora los bloques que permiten el prototipado rápido de sistemas de control (Figura 2.36) y permite la implementación de algoritmos de control compatibles con la plataforma de experimentación.

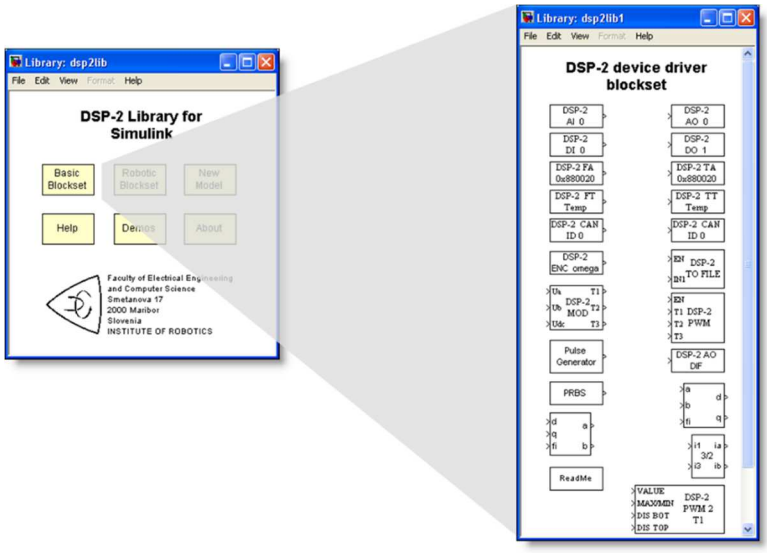

Figura 2.36. Bloques incluidos en la librería Simulink para el control del entorno de desarrollo DSP-2 implementado por la Facultad de Ingeniería Eléctrica y Ciencias de la Computación de la Universidad de Maribor.

Una vez el alumno desarrolla su sistema de control y comprueba su funcionamiento mediante el simulador incluido en la herramienta Simulink, puede acceder al laboratorio remoto para llevar a cabo la programación del DSP. Antes es necesario realizar la reserva de la instancia de experimentación deseada.

Cuando accede a la web del experimento, selecciona el fichero binario generado mediante Simulink y accede al panel de control implementado mediante instrumentos virtuales de LabView para llevar a cabo la monitorización y control del experimento. Paralelamente puede acceder a la webcam que monitoriza la instancia de experimentación utilizada.

De esta forma la experimentación llevada a cabo mediante el laboratorio remoto es análoga a la que se desempeña en los laboratorios presenciales.

2.12.4 Resumen

Este laboratorio remoto permite llevar a cabo experimentación con sistemas de control implementados mediante un procesador digital de señales. Los desarrolladores han conseguido remotizar un experimento que previamente han desarrollado en laboratorios presenciales manteniendo intactas los principales procesos requeridos para esta experimentación.

El laboratorio remoto se fundamenta en permitir la experimentación con el sistema de desarrollo DSP-2 mediante el empleo de LabView para la monitorización y control del experimento, si bien este hecho se mantiene del experimento original en los laboratorios presenciales. Esta herramienta facilita la publicación en Internet mediante la aplicación LabView web Publishing, cuyo único requerimiento por el lado del cliente es un navegador con el plugin de LabView.

Por otro lado este laboratorio requiere la instalación de MATLAB/Simulink en el ordenador cliente. El elevado coste de la licencia requerida limita enormemente la posibilidad de acceder desde ordenadores personales de los estudiantes fuera de los laboratorios de la Universidad.

2.13 REAL, Remote Engineering and Applications Laboratory de la Universidad Tecnológica de Illmenau

Este laboratorio remoto ha sido desarrollado por el Grupo de Sistemas de Comunicación Integrados en la Universidad Técnica de Illmenau bajo la coordinación del Dr. Karsten Henke (Henke, Tabunshchyk, Wuttke, Vietzke, & Ostendorff, 2014) (Henke, Ostendorff, Wuttke, & Vogel, 2012) (Hanke, Ostendorff, & Wuttke, 2012). El laboratorio remoto REAL es un sistema interactivo híbrido donde los estudiantes pueden diseñar un algoritmo de control para controlar mediante distintas tecnologías (Figura 2.37) un conjunto de maquetas electromecánicas.

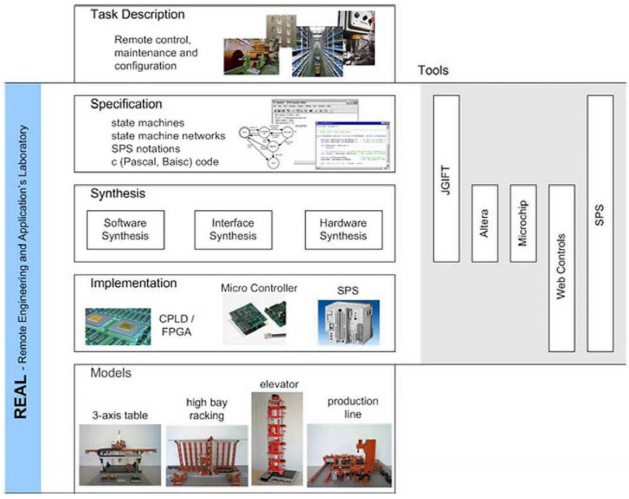

Figura 2.37. Esquema general del laboratorio remoto REAL de la Universidad Técnica de Illmenau.

2.13.1 Funcionalidad

Se trata de un laboratorio remoto híbrido porque permite experimentar directamente con las maquetas electromecánicas o proporcionando modelos de simulación de los sistemas físicos.

Este laboratorio proporciona los recursos de monitorización y control necesarios para permitir observar y probar todas las propiedades del diseño. Siguiendo las metodologías de diseño formales, el laboratorio permite ejecutar primero una simulación e incluso la creación de prototipos para establecer una base previa al desarrollo de un diseño de sistema fiable. Además, el sistema REAL incluye recursos de simulación para validar la funcionalidad de todo el diseño. Para ello, desde un interfaz gráfico de usuario basado en web, se ofrecen múltiples posibilidades para la ejecución de simulaciones, tales como:

- Uso de modelos de simulación del sistema físico para la creación de prototipos.
- Depuración de programas paso a paso y ejecución en paralelo de varios prototipos.
- Visualización del proceso de simulación con las herramientas.
- Características de generación de patrón de prueba.
- Generación de código para el hardware y el software de síntesis.

2.13.1.1 Integración con otras plataformas de aprendizaje o experimentación

Aunque le laboratorio REAL puede funcionar de manera autónoma toda su infraestructura ha sido desarrollada en base a la arquitectura iLab, lo cual permite su interconexión con iLab y otros RLMS (Sahara, WebLab-Deusto, etc…).

Además el laboratorio REAL ha sido integrado con el sistema de gestión del aprendizaje de la Universidad Técnica de Illmenau, adecuándose a su sistema de autenticación e incluyendo un método de validación automática de los diseños

realizados por los estudiantes que permite al LMS llevar un seguimiento paso a paso de las tareas ejecutadas por los estudiantes (Figura 2.38).

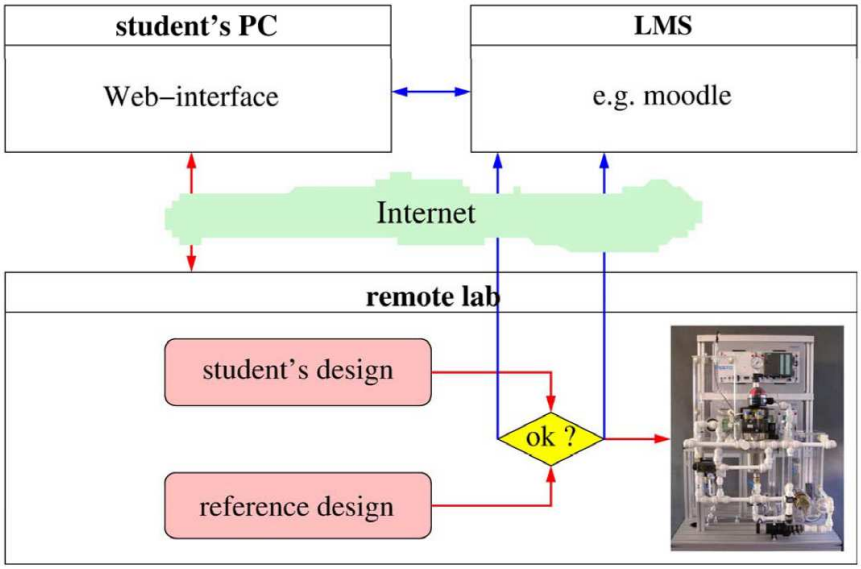

Figura 2.38. Validación del diseño del estudiante bajo el control del LMS en el laboratorio REAL de la Univesidad Tecnológica de Illmenau.

2.13.1.2 Escalabilidad

La capacidad de permitir al estudiante experimentar dentro del laboratorio remoto REAL sobre distintas maquetas electromecánicas con distintas tecnologías exige la interconexión de todas las plataformas de experimentación con los dispositivos finales mediante un bus denominado "Bus de laboratorio remoto interno" (internal remote lab bus), que permite añadir tanto nuevas plataformas de experimentación, con independencia de la tecnología (microcontroladores, dispositivos lógicos programables, PLC, etc…), como dispositivos finales (Figura 2.39).

Este mismo bus, capaz de interconectar todas las plataformas de experimentación con los diferentes dispositivos, impide el acceso concurrente del laboratorio soportando una única sesión simultánea en la que sólo se podrá establecer una configuración de una plataforma de experimentación conectada a un dispositivo.

2.13.1.3 Desplegabilidad

La complejidad de la arquitectura implementada por este laboratorio y el alto coste del equipamiento utilizado complica el despliegue de este laboratorio en otras instituciones.

Todos los componentes hardware han sido desarrollados a medida para el laboratorio y son propiedad del equipo desarrollador.

2.13.1.4 Replicabilidad

La propia arquitectura del laboratorio (Figura 2.39) está diseñada para permitir la experimentación con múltiples tecnologías. Actualmente el estudiante puede seleccionar entre un microcontrolador PIC, una FPGA de Altera o un PLC de Beck IPC como plataforma de experimentación. Así mismo, la versatilidad del "bus de laboratorio remoto interno" permite al estudiante seleccionar entre diversas maquetas sobre las que efectuar la experimentación.

Sin embargo, para cada plataforma de experimentación o dispositivo a controlar que se quiera añadir al laboratorio es necesario desarrollar respectivamente una "unidad de protección del bus" o una "unidad de protección física del sistema" específica que se implementa mediante un dispositivo lógico programable en el que se establecen las restricciones de cada dispositivo. Esto complica enormemente la adición de nuevas tecnologías.

2.13.1.5 Disponibilidad

El acceso al laboratorio se lleva a cabo desde el propio LMS de la Universidad Tecnológica de Illmenau. Los estudiantes deben reservar el laboratorio durante una fracción de tiempo en la que dispondrá de acceso exclusivo al laboratorio. Esto representa una limitación del laboratorio debido a que con independencia del número de plataformas de experimentación y maquetas electromecánicas integradas en el laboratorio, el acceso al mismo es exclusivo a un único estudiante, pudiendo generar problemas de disponibilidad cuando se emplea en cursos con un número considerable de estudiantes.

2.13.1.6 Conectividad

Este laboratorio no requiere de ninguna configuración específica en la infraestructura de comunicaciones del cliente. Únicamente es necesario Internet para conectarse con el servidor web que proporciona el cliente a través del puerto 80.

2.13.1.7 Universalidad

Pese a que toda la experimentación se desarrolla desde un navegador web. Los diferentes interfaces de control de cada experimento incluido están desarrollados mediante applets de Java (todos ellos sin firmar) lo cual dificulta el acceso desde terminales móviles e incluso desde muchos navegadores web.

2.13.2 Implementación

La arquitectura del laboratorio remoto REAL puede verse en la Figura 2.39.

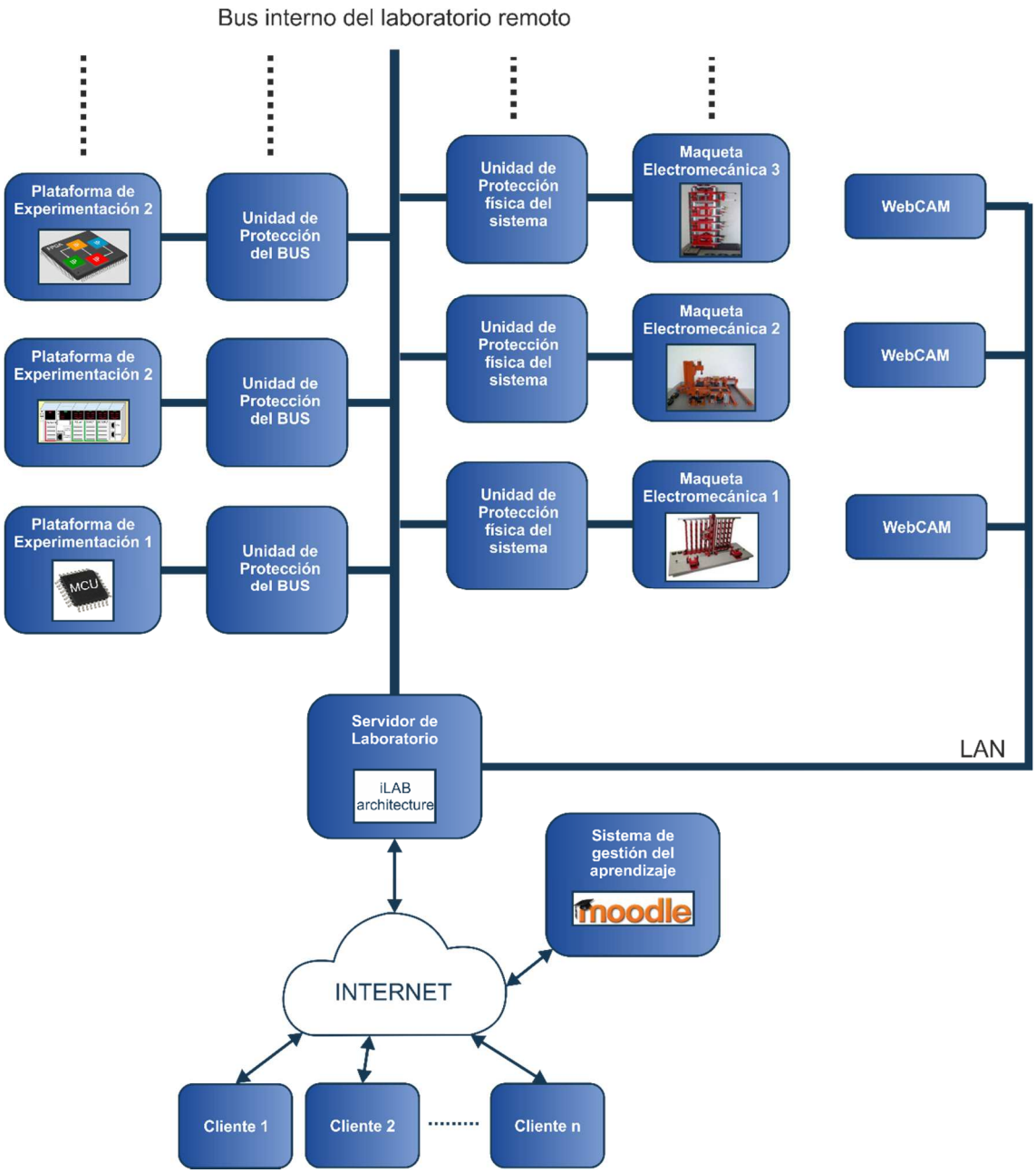

Figura 2.39. Arquitectura del laboratorio REAL desarrollado por la Universidad Tecnológica de Illmenau.

La principal característica diferenciadora de este laboratorio remoto es que posibilita la experimentación desde diferentes plataformas de experimentación para el control de distintos dispositivos, permitiendo al estudiante configurar en cada sesión los recursos a utilizar en la misma. Tres componentes de la arquitectura han sido desarrollados para proporcionar esta flexibilidad de una forma segura:

- El "bus interno del laboratorio remoto" que permite interconectar todas las partes del laboratorio.
- La "unidad de protección de bus" cuya función es interconectar las diferentes plataformas de experimentación al bus laboratorio remoto, protegiéndolo de bloqueo, mal uso y daños.

- La "unidad de protección del sistema físico", que protege las maquetas electromecánicas integradas en el laboratorio remoto contra daños deliberados o secuencias de control equivocadas accidentalmente.

2.13.2.1 Servidor de laboratorio

El servidor del laboratorio puede realizar las tareas de autenticación y gestión de reservas, aunque en la implementación actual las principales tareas de administración han sido transferidas al LMS. Actualmente las principales funciones de este componente son las de ejercer de servidor web de todos los interfaces que permiten la interacción con los experimentos, enviar los diseños realizados por los estudiantes a las unidades de protección del bus y permitir la interacción online mediante un conjunto de señales gestionadas mediante una tarjeta de entradas y salidas que está conectada al bus interno del laboratorio remoto.

2.13.2.2 Plataformas de experimentación

La arquitectura de este laboratorio remoto permite la adecuación de cualquier plataforma de desarrollo, independientemente de la tecnología embebida en la que se base.

Esto es así porque todas las entradas y salidas se interconectan al bus interno del laboratorio remoto, filtradas mediante una unidad de protección del bus que adecua los niveles entre los diferentes sistemas de desarrollo. Además, esta unidad lleva a cabo la programación del dispositivo mediante un programador integrado.

2.13.2.3 Unidades de protección de bus

Para cada plataforma de experimentación que se integre en el laboratorio es necesario desarrollar a medida una unidad de protección de bus que permita la interconexión con el bus interno del laboratorio remoto, así como la programación del dispositivo a partir del diseño recibido desde el servidor del laboratorio.

La unidad de protección del bus recibe órdenes de una unidad de control y comprueba la validez de las mismas antes de volcarlas al bus. Esto se hace mediante la verificación del protocolo de transporte. El contenido de los datos y direcciones de transmisión no se comprueban, porque esto depende del sistema físico utilizado y por lo tanto es llevado a cabo por la unidad de protección específica de cada sistema físico. La función de esta unidad es evitar que una unidad de control pueda erróneamente bloquear el bus y evitar que otros experimentos se vean afectados. La unidad de protección de bus se basa en el mismo hardware que las unidades de protección para los sistemas físicos en el laboratorio remoto pero utiliza una serie de componentes diferentes para simplificar la producción y mantenimiento.

Una tarea secundaria para la unidad de protección del bus es interconectar varias unidades de control al bus. Una de las principales limitaciones de este laboratorio remoto es que estas unidades sólo pueden generar entradas y salidas digitales simples para interactuar directamente con los sistemas físicos, impidiendo por lo tanto utilizar protocolos o señales analógicas en el control de dispositivos.

Por el contrario, una ventaja de utilizar una unidad de protección del bus es que entre sus funciones está la de adaptar el nivel de la plataforma de experimentación con la tensión del bus.

Finalmente, la unidad de protección del bus actúa como programador para las plataformas de experimentación integradas en el laboratorio. Recibe los archivos de programación desde el cliente web a través del servidor de laboratorio.

2.13.2.4 Unidad de protección de los sistemas físicos

La principal función de este componente es proteger al dispositivo a controlar permitiendo que los estudiantes puedan ejecutar sus algoritmos directamente en la unidad de control siendo totalmente libres en la elección de las herramientas de desarrollo. La unidad de protección del sistema físico comprueba la del diseño del estudiante mediante el filtrado de todas las señales que son enviadas al dispositivo. Sólo aquellas señales que no va a causar ningún mal funcionamiento se vuelcan al dispositivo. Todas las demás se descartan y opcionalmente presentan una condición de error a un sistema de gestión de aprendizaje (LMS). Las señales pueden ser enviadas desde la plataforma de experimentación elegida o directamente desde el cliente web, a través de la tarjeta de entradas integrada en el servidor del laboratorio y conectada al bus interno del laboratorio remoto (Figura 2.40).

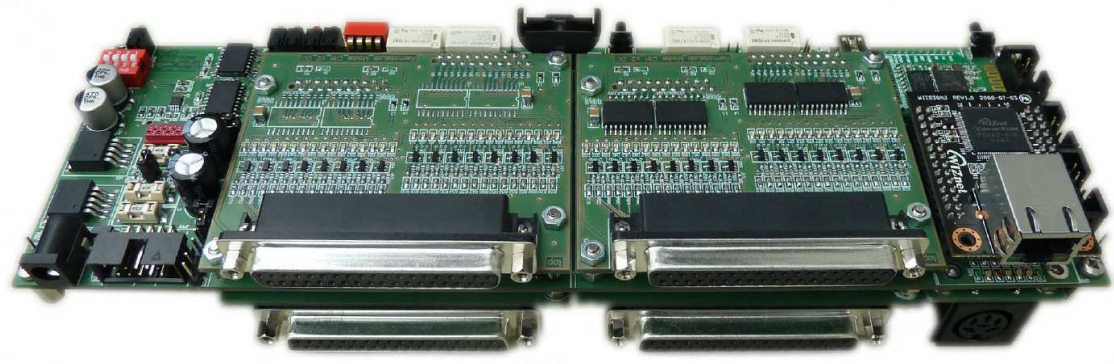

Figura 2.40. Plataforma hardware utilizada para implementar la unidad de protección del sistema físico en la laboratorio REAL desarrollado por la Universidad Tecnológica de Illmenau.

Este concepto se puede utilizar con todas las unidades de control como microcontrolador, FPGA, PLC, etc. El uso de una unidad de protección tan universal da

a los estudiantes el mayor grado de libertad para su diseño, ya que no hay precauciones que tengan que ser tenidas en cuenta.

2.13.2.5 Cliente

Como ya se ha comentado anteriormente toda la experimentación del estudiante se lleva a cabo desde un navegador por medio de la web alojada en el servidor del laboratorio. Las distintas páginas web que permiten interactuar con las diferentes maquetas integradas en el laboratorio están desarrolladas mediante applets de Java no firmados que dificultan el acceso desde ciertos navegadores.

2.13.3 Escenario de uso

El enfoque de este laboratorio remoto es permitir al estudiante llevar a cabo experimentación a través de Internet de la misma forma en la que lo haría en un laboratorio presencial. La idea es que cuando el estudiante accede al laboratorio debe elegir los equipos con los que va a experimentar e implementar un prototipo para validar el diseño desarrollado.

Cuando el estudiante comienza una sesión con el laboratorio, previamente debe haber realizado la reserva, accede a una web de configuración en la cual debe elegir la plataforma de experimentación y el dispositivo (maqueta electromecánica) que desea controlar. Existen también recursos virtuales que permiten al estudiante realizar simulaciones para probar el diseño antes de implementarlo físicamente.

Cuando el estudiante ha montado su experimento, el siguiente paso es seleccionar el fichero binario con el que programar la plataforma de experimentación. Este fichero es completamente independiente al laboratorio remoto y debe haber sido desarrollado por el estudiante utilizando el entorno de desarrollo por él preferido.

Cuando el estudiante da la orden de programar el dispositivo, el servidor del laboratorio recibe el fichero mediante un comando POST y lo transfiere a la unidad de protección del bus asociada a la plataforma de experimentación seleccionada, que se encarga de llevar a cabo la programación de esta.

En ese momento se carga una web con un interfaz gráfico de usuario desarrollado por medio de un applet de Java para permitir la experimentación sobre el dispositivo conectado. Existen dispositivos que no requieren experimentación, mientras que otros requieren de un interfaz de usuario que permita enviar órdenes durante la experimentación. Incluso pueden existir casos en los que la maqueta electromecánica no requiere un control mediante un sistema embebido y la experimentación se desarrolla directamente desde los controles dispuestos en el interfaz de usuario.

2.13.4 Resumen

Dado que la funcionalidad del experimento es controlar maquetas electromecánicas didácticas, que son equipos de alto coste y en los laboratorios presenciales sufren los errores de los estudiantes y requieren un continuo mantenimiento, se ha puesto un especial cuidado en impedir que el estudiante pueda dañar el dispositivo a controlar para dotar de integridad y sostenibilidad al laboratorio remoto. El filtrado de las señales ejecutada por las unidades de protección del bus y de protección de los sistemas físicos hace que este laboratorio remoto esté a prueba de errores e incluso de ataques malintencionados. Esta característica elimina los miedos que generalmente asaltan a los estudiantes cuando se disponen a trabajar con equipamiento de alto coste, suponiendo una ventaja incluso sobre los laboratorios presenciales.

Sin embargo el uso de estos dispositivos de filtrado imponen un alto coste al laboratorio remoto puesto que únicamente permiten el control de dispositivos mediante simples líneas digitales. La proliferación de sensores y dispositivos analógicos o controlados mediante buses de comunicación (I2C, SPI, CAN , etc…) limitan enormemente la experimentación.

Este sistema tiene como característica fundamental proporcionar al estudiante un laboratorio único para el desarrollo de aplicaciones embebidas dirigidas al control de sistemas electromecánicos sin limitaciones en cuanto a la tecnología ni al dispositivo controlado. Esta capacidad de configuración del experimento por el estudiante, conseguida mediante el bus interno del laboratorio remoto, impone un acceso exclusivo un usuario lo cual limita su adecuación a cursos con alto número de estudiantes.

En lo referente al coste del laboratorio, la utilización de maquetas industriales y la necesidad de diseñar circuitos a medida para cada tecnología de desarrollo y sistema a controlar elevan su coste muy por encima del calculado en el resto de sistemas analizados.

2.14 Otros Laboratorios Remotos para la experimentación con sistemas embebidos

Además de estos diez laboratorios remotos que han sido analizados, existen otros sistemas reseñables que no han sido detallados bien por similitud con alguno de los incluidos anteriormente o por falta de disponibilidad durante el desarrollo de este capítulo. Sin embargo, para consolidar el mapa actual de laboratorios remotos para la experimentación con sistemas embebidos, los principales sistemas desplegados han sido incluidos en la Tabla 2.2 que recoge sus principales características. El desglose de laboratorios remotos contemplados en esta comparativa se incluye a continuación:

1.- Laboratorio remoto para FPGA de la Universidad de Erlangen-Nuremberg (Reichenbach M. , Schmidt, Pfundt, & Fey, 2011).

2.- Laboratorio remoto para FPGA de la Universidad de Coimbra (Lobo, 2011).

3.- "ViciLab" un Laboratorio Remoto para experimentación con circuitos digitales y FPGA desarrollado por la Universidad Nacional de Irlanda en Galway (Morgan, y otros, 2014).

4.- Labshare FPGA Rig (Lowe, Murray, Lindsay, & Liu, 2009).

5.- CPLD Hybrid Lab de la Universidad de Ciencias Aplicadas de Carintia (Pop, Zutin, Auer, Henke, & Wuttke, 2011).

6.- AT89S52 MCU Remote lab de la Universidad de Indonesia (Limpraptono, Ratna, & Sudibyo, 2012).

7.- Laboratorio de educación a distancia, MicroLab de la Universidad Demirel Sulayman (Kutlu & Taşdelen, 2010).

8.- Arduino Remote lab en la Universidad Abierta Helénica (HOU) (Fotopoulos, Anastasios, & Anastasios, 2013).

9.- Laboratorio remoto para basado en un DSP de la Universidad de Maribor (Hercog, Gergic, Uran, & Jezernik, 2007).

10.- REAL, Remote Engineering and Applications Laboratory de la Universidad Tecnológica de Illmenau (Henke, Ostendorff, Wuttke, & Vogel, 2012).

11.- Laboratorio remoto para FPGA de la Universidad de Ciencia y Tecnología de Hanoi (Hoang, y otros, 2015).

12. eDiViDe: Laboratorio virtual Europeo para el diseño de sistemas digitales (Vandorpe, y otros, 2013).

13.- Laboratorio remoto para FPGA de la Universidad de Jiangsu (Hui & Tie-jun, 2008).

14.- Laboratorio Online de sistemas digitales de la Universidad de Makerere (Butime, y otros, 2012).

15.- Laboratorio remoto para FPGA de la Universidad de Vadapalani (Karthik, Shreya2, & Srihari, 2014).

16.- Laboratorio remoto para MCU 80C537 de la Universidad politécnica de Cataluña y la Universidad de Ciencias Aplicadas de Carintia (Gilibert, Picazo, Auer, Pester, & Cusidó, 2006).

17.- Laboratorio RexPIC del Instituto Federal de Ceará (Goncalves de Moraes & Mendes de Sales, 2012).

18.- Laboratorio remoto para diseño de sistemas basados en microcontroladores de la Universidad de Rijeka (Zenzerović & Sučić, 2011).

19.- Laboratorio DistanceLab de la Universidad Tecnológica de Tallin. (Sell, y otros, 2015).

20.- Laboratorio remoto para experimentación con microcontroladores de la Universidad de Pamplona (Colombia) (Rangel, Chacón, & Araque, 2011).

21.- Laboratorio de experimentación remota con FPGA de la Universidad de Fayoum (El-Medani, 2008).

22.-Laboratorio Remoto para microcontroladores de 8 bits de la Universidad de Oporto y la Universidad de Australia del Sur (Ferreira, Nedić, Machotka, Nafalski, & Göl, 2010).

23.- Open FPGA Remote Lab de la Universidad Nacional de Educación a Distancia (Tawfik, Sancristobal, Martin, Diaz, & Castro, 2012).

24.- PICs Remote Lab de la Universidad Nacional de Educación a Distancia (Tawfik, Sancristobal, Martin, Diaz, & Castro, 2012).

25.- Laboratorio para experimentación con Arduino de la Universidad Técnica Nacional de Zaporizhzhya (Anzhelika, Olga, Ivanov, Sokolyanskii, & Kurson, 2015).

Tabla 2.2. Análisis de características fundamentales de los principales laboratorios remotos para la experimentación con sistemas embebidos

	Integración	Escalabilidad	Desplegabilidad	Replicabilidad	Disponibilidad	Conectividad	Universalidad	Arquitectura	Interacción
1 FPGA FAU	Stand-Alone	10 instancias. *Cada instancia de experimentación conectada por USB con el servidor de laboratorio.*	Sw: Open Source Hw: Propietario	Compatible dispositivo programables por JTAG	Alumnos VHB	TCP 22, 8001-8010, 8101-8110 (2x instancia)	Clte SSH, Navegador web (Plugin VLC) Xilinx webpack ISE	Mixta	M:Virtual I: E.Digitales
2 FPGA UC	Stand-Alone	Instancia única	Sw: Quartus II, YawCam Hw: Comercial	Compatible con FPGAs de Altera familia Cyclone	Abierta	HTTP:80	Navegador web (Java applet no firmado) Quartus II	Centralizada	M:WebCAM I: E.Digitales
3 ViciLab	Plataforma ViciLogic (LMS Propietario)	16 instancias *Instancias de experimentación interconectadas en torres de 8 por Ethernet*	Sw: Propietario Hw: Propietario	Universal para FPGA	Abierta	HTTP:80	Aplicación ViciLab (propietaria) PlanAhead ó Quartus II	Distribuida	M: Virtual I: E.Digitales
4 Labshare FPGA	Sahara RLMS	4 x Instancias *Cada instancia de experimentación se conecta por cable paralelo a un PC para programación.*	Sw: Abierto Hw: Abierto	Compatible con FPGAs de Xilinx familias Spartan y Virtex	Red Labshare	HTTP:80	Navegador web Xilinx webPack ISE	Replicada	M: webcam I: E.Digitales, Gen. Func
5 FPGA CUAS	RLMS iLab	Instancia única	Sw: MAxPLUS, Citrix y Labview VI Hw: Propietaria	Compatible con CPLDs de Altera familia MAXPLUS	Lab2Go	TCP 135	Navegador web (applet de Java de Citrix y Plugin Labview) Altera MAX+PLUS	Centralizada	M: Virtual I: E.Digitales
6 MCU UI	Stand-Alone	2 instancias *Las instancias de experimentación se conectan mediante un*	Sw: Propietario Hw: Propietario	Compatible con plataformas basadas en MCU atmel AT89S52	Por invitación – Durante impartición de curso en Universidad de Indonesia	HTTP:80	Navegador web IDE para atmel AT89S52	Distribuida	M: webcam I: E.Digitales

		convertidor Ethernet/serie					
7 MicroLab SDU	Stand-Alone	3 instancias *Las instancias de experimentación se interconectan mediante Bus CAN.*	Sw: Propietario He: Propietario	Compatible con MCU reprogramables con UART integrado	Abierta	TCP 2000-2020	Aplicac Client (Compil
8 Arduino HOU	Stand-Alone	Instanca Única	Sw: Abierto Hw: Abierto	Compatible con plataformas Arduino	Abierta	HTTP:80	Navega (applet Flash
9 DSP UM	LMS Moodle	4 Instancias *Cada instancia requiere un servidor de laboratorio. No hay balanceo de carga.*	Sw: Propietario Hw: Propietario	Plataforma DSP-2	Alumnos Facultad de Ingeniería Eléctrica y Ciencias de la Computación de la Universidad de Maribor	HTTP:80	Navega (Plugin Matlab/
10 REAL ITU	LMS Moodle RLMS Shara	(3) x (3) Instancias *Cada plataforma de experimentación y dispositivo se conecta al Bus del laboratorio.*	Sw: Propietario Hw: Propietario	Universal	Alumnos Universidad Tecnológica de Illmenau	HTTP:80	Navega (applets IDE par seleccio
11 FPGA HUST	Stand-Alone	2 Instancias *Cada instancia de experimentación conectada por USB con el servidor de*	Sw: Propietario Hw: Servidor de Interacción propietario	Compatible con FPGAs de Altera familia Cyclone	Alumnos Universidad de Hanoi	TCP: 2000-2001	Aplicac escritor platafor Windov (propie

13 FPGA (Jiangsu)	Stand Alone	Instancia Única	Sw: Propietario Hw: Propietario	Palataformas de experimentación basadas en FPGA compatibles JTAG	Alumnos Universidad de Jiangsu	TCP:2001	Aplicac escritor platafor Windov IDE con FPGA
14 FPGA UM	RLMS iLab	Instancia Única	Sw: Comercial (LabView Vi) Hw: Comercial	NI FPGA Digital Electronics laboratory	Alumnos Universidad de Jiangsu	HTTP	Navega (Plugin
15 FPGA SRMU	Stand Alone	Instancia Única	Sw: Propietario Hw: Comercial	FPGA Spartan 3	Alumnos Universidad de Makerere	HTTP	Navega (Plugin
16 MCU UPC	RLMS iLab	Instancia Única	Sw: MAxPLUS, Citrix y Labview VI Hw: Propietaria	Infineon SAB 80C537	Lab2Go	TCP 135	Navega (applet Citrix y Labviev Altera M
17 RexPIC IFCE	Stand Alone	Instancia Única	Sw: Microchip TCPIP stack Hw: Propietario	Familia MCU Microchip PIC 16F	Abierta	HTTP	Navega Microch IDE
18 PIC UR	Stand Alone	Instancia Única	Sw: Propietario Hw: Comercial	Familia MCU Microchip PIC 16F	Obsoleto	HTTP	Navega Microch IDE
19 DistanceLab	DistanceLAB	Varios Laboratorios	Sw: Propietario	MCU ATmega2561-	Abierta	HTTP	Navega

20 PIC IIDTA	Stand Alone	Instancia Única	Sw: Propietario Hw: Propietario	Familia MCU Microchip PIC 18F	Alumnos de la Universidad de Pamplona (Colombia)	HTTP	Navega (Plugin Microch IDE
21 FPGA UF	Stand Alone	20 Instancias *Cada instancia dispone de su servidor de laboratorio. Servidor de administración común con balanceo de carga.*	Sw: Propietario Hw: Comercial	FPGA Xilinx Spartan 3	Alumnos de la Universidad de Fayoum (Egipto)	TCP: 8000-8020	Aplicac escritor platafor Windov Xilinx w
22 MCU UP	LMS Moodle	Instancia Única	Sw: Propietario Hw: Propietario + NI Elvis	MCU 80C51	Alumnos de la Universidad de Oporto y Universidad de Australia del Sur	HTTP	Navega (Plugin Keil uV
23 FPGA UNED	Stand Alone	Instancia Única	Sw: Abierto Hw: Comercial	Compatible con FPGAs de Xilinx familias Spartan y Virtex	Alumnos UNED	TCP:90 RDP	Navega (Plugin
24 PIC UNED	Stand Alone	Instancia Única	Sw: Abierto Hw: Comercial	Familia MCU Microchip PIC 18F	Alumnos UNED	TCP:90 RDP	Navega (Plugin
Arduino (Ukrania)	Stand-Alone	4 Instancias *Cada instancia de experimentación*	Sw: Abierto Hw: Abierto	Compatible con plataformas Arduino	Abierta	HTTP:80	Navega (applet Flash

2.15 Consideraciones finales

Una vez analizado un conjunto significativo de laboratorios remotos, podemos extraer una serie de consideraciones que deberán ser consideradas al implementar una nueva arquitectura.

Estudios desarrollados sobre arquitecturas de laboratorios remotos multidisciplinares (Lowe, Murray, Lindsay, & Liu, 2009) remarcan dos objetivos fundamentales que deben priorizarse: facilitar el intercambio de recursos entre diferentes instituciones (1) y reducir el coste de la experimentación (2). En el caso específico de los laboratorios remotos destinados a la experimentación con sistemas embebidos los objetivos difieren debido a que generalmente el coste de las plataformas de experimentación no es elevado.

La rápida evolución de las tecnologías utilizadas para la implementación de sistemas embebidos (Figura 2.41) exige la permanente asimilación de nuevos plataformas tecnológicas y requiere la actualización continua de los laboratorios, algo que resulta especialmente complicado para instituciones educativas que no pueden permitirse el reemplazo continuo de los sistemas de desarrollo utilizados en los laboratorios presenciales. Los laboratorios remotos permiten a los estudiantes experimentar con plataformas de última generación y puede resultar un complemento perfecto para los experimentos llevados a cabo presencialmente en los laboratorios.

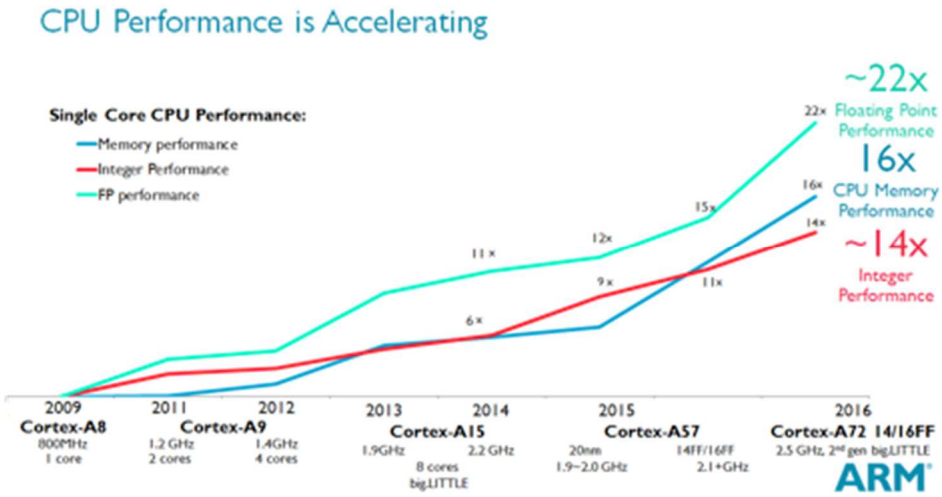

Figura 2.41. Evolución de modelos y rendimiento de la familia A de ARM en los últimos 7 años.

Por otro lado el diseño de sistemas embebidos exige llevar a cabo un diseño conjunto del motor hardware y de la arquitectura software de aplicación (Yen & Wolf, 1996),

siendo esta última parte la más determinante en el rendimiento del sistema final (Joshi & Gurumurthy, 2014). Los laboratorios remotos permiten completar las fases del diseño software con independencia del hardware, acelerando el proceso de asimilación de los contenidos y ayudando a optimizar el tiempo de experimentación el laboratorios presenciales (de Jong, Linn, & Zacharia, 2013).

Como se ha podido observar tras el análisis de los laboratorios seleccionados, existen sistemas que implementan de manera independiente todas las tareas de administración y otros que se integran en sistemas de gestión de laboratorios remotos (iLab, Sahara, webLab-Deusto, etc…) o sistemas de gestión del aprendizaje (Moodle, Sakai, Blackboard, etc…) (San Cristóbal, 2010), delegando gran parte de estas tareas de administración, como autenticación, reserva, trazabilidad de sesiones, etc… sobre estas plataformas, en las que la comunidad que las soporta garantiza su mantenimiento y evolución. Además existen soluciones combinadas en las que el administrador encargado de la instalación del laboratorio puede elegir entre integrar cada uno de los módulos en estas plataformas o desplegar un sistema independiente que las soporte. Esta última opción facilita el despliegue y la adecuación a los requerimientos de las diferentes instituciones educativas.

Debido a la exigencia de programación física de las plataformas de experimentación (microcontroladores, FPGAs, DSPs, etc…) en los laboratorios remotos destinados a la experimentación con sistemas embebidos no es posible el acceso simultáneo de múltiples usuarios a una misma instancia. Entre los laboratorios analizados existen sistemas que permiten el acceso concurrente de múltiples estudiantes pero, por la naturaleza de la tecnología, únicamente se permite la experimentación con el diseño de un estudiante que en estos casos, toma el papel de conductor de la experimentación y relega la capacidad del resto de usuarios a la monitorización e interacción con su propio diseño (Kutlu & Taşdelen, 2010). Por tanto, otra característica fundamental que debe cumplir un laboratorio remoto es la escalabilidad, permitiendo el despliegue de varias instancias de experimentación entre las que se balancee el acceso de los usuarios. Esta característica es prioritaria si se desea utilizar un laboratorio remoto con dominios importantes de estudiantes.

Otra característica crítica para un laboratorio remoto es facilitar la conectividad con el mismo desde distintas instalaciones. Teniendo en cuenta que los laboratorios remotos son comúnmente accedidos desde instituciones educacionales en las que generalmente la conexión se lleva a cabo por detrás de un cortafuegos, utilizar puertos poco comunes puede provocar errores de acceso que impliquen a los servicios IT llevar a cabo configuraciones específicas que vulneran la seguridad de la instalación.

En este sentido, es obvio que una de las principales ventajas de los laboratorios remotos es que permiten el acceso por los estudiantes a cualquier horario y desde cualquier ubicación. Esta universalidad se ve mermada si para el acceso se impone la utilización de aplicaciones específicas que están implementadas solo en algunas plataformas o requieren la ejecución applets o Plugins que únicamente son compatibles con ciertos navegadores. Utilizar clientes soportados por un navegador web mediante el uso de tecnologías como HTML5, JavaScript y XML (AJAX) permite simplificar las arquitecturas del laboratorio remoto, y puede proporcionar un entorno más integrado y sensible que además de permitir a los estudiantes acceder desde dispositivos móviles (Orduña P. , García-Zubia, Irurzun, & Rodriguez-Gil, 2011) (Garcia-Zubia, Orduna, Lopez-de-Ipina, & Alves, 2009).

Desde el punto de vista del desarrollo de la experimentación, es importante contemplar cómo se va a desarrollar la experimentación por parte del estudiante. Muchos laboratorios permiten contemplar las tareas de programación y validación, pero exigen que la codificación y compilación o síntesis sean desarrolladas por el estudiante desde su propio dispositivo y mediante el empleo de sistemas de desarrollo integrados que generalmente están proporcionados por el fabricante de los dispositivos en los que se fundamenta la plataforma de experimentación. Esto limita enormemente el tiempo de acceso al laboratorio ya que las tareas del ciclo de diseño que más tiempo consumen son desarrolladas sin ocupar el laboratorio. Sin embargo, utilizar tecnologías web compatibles con los principales navegadores no sirve de nada si la experimentación requiere el empleo de un sistema de desarrollo que únicamente está disponible para ciertas plataformas, ya que poder acceder al laboratorio para programar la plataforma de experimentación y validar el diseño mediante interacción y monitorización no sirve de nada si ese diseño no puede ser corregido o modificado. Otros laboratorios incluyen un editor integrado desde el que el estudiante puede codificar el diseño y herramientas para generar el fichero binario resultante. Por este motivo el tiempo necesario para la experimentación en estos laboratorios es muy grande y los sistemas de colas no resultan válidos, siendo necesario utilizar un sistema de reservas que permite a los estudiantes reservar el acceso al laboratorio durante un tiempo proporcional al requerido por la experimentación. Estos sistemas pueden generar problemas de disponibilidad en ciertos periodos (p.e. entrega de trabajos) si el número de instancias disponibles no es acorde al dominio de estudiantes que acceden al laboratorio.

En definitiva, tras analizar los principales laboratorios remotos desplegados para el desarrollo de experimentación con tecnologías propias de los sistemas embebidos, se pueden extraer varias conclusiones específicas que determinan el rendimiento de estos sistemas en particular, no alineadas con los resultados publicados en la bibliografía de laboratorios remotos genérica (Lowe, Murray,

Lindsay, & Liu, Evolving Remote Laboratory Architectures to Leverage Emerging Internet Technologies, 2009) (de Jong, Linn, & Zacharia, 2013) (otros)

- Las características particulares de los sistemas embebidos exigen la disposición de múltiples instancias para la experimentación concurrente de múltiples estudiantes. *Por tanto la escalabilidad de los laboratorios remotos para experimentación con tecnologías embebidas se convierte en la característica más determinante que condiciona el rendimiento de estos sistemas.*

- La continua evolución de las diferentes tecnologías embebidas lleva a limitar el ciclo de vida de los nuevos dispositivos embebidos a una ventana que oscila entre los 5 y los 10 años en función de los distintos fabricantes. Además teniendo en cuenta la finalidad didáctica de los laboratorios remotos es necesario la continua actualización las tecnologías sobre las que se desarrolla la experimentación para la formación de estudiantes con las competencias demandadas por la industria. *Por ello, la replicabilidad de los laboratorios remotos desarrollados para experimentación con tecnologías embebidas, se torna una característica fundamental para garantizar la sostenibilidad de los laboratorios.*

- A diferencia de otros laboratorios remotos que permiten la experimentación con equipos de otras disciplinas, el bajo coste de los sistemas de desarrollo embebidos propicia que *son los dispositivos requeridos para la remotización del laboratorio los que más determinantemente influyen en el coste total de los laboratorios remotos* para la experimentación con sistemas embebidos.

La Tabla 2.2 demuestra una dependencia total entre las arquitecturas adaptadas por cada laboratorio remoto y las características citadas que se puede resumir en de la siguiente forma:

- ***Arquitecturas Centralizadas***: Basadas en proporcionar acceso remoto sobre las plataformas de experimentación comerciales, impiden la escalabilidad del sistema o limitan enormemente el número de instancias posible. Permiten la replicabilidad, permitiendo la sustitución de los diferentes componentes que conforman la instancia de experimentación.
- ***Arquitecturas Distribuidas***: Se fundamentan en la experimentación que va a ser provista por el laboratorio. Facilitan la escalabilidad de los laboratorios pero en la mayoría de los sistemas presentan una dependencia fuerte con la tecnología específica presente en la plataforma de experimentación que complica la replicabilidad.
- ***Arquitecturas Mixtas***: Condicionan la escalabilidad a componentes de las instancias que se conectan directamente con el servidor del laboratorio.
- ***Arquitecturas Replicadas***: Permiten la escalabilidad a fuerza de replicar el servidor del laboratorio generando un alto coste de la instalación.

Capítulo

3

Especificación de la arquitectura propuesta

Demostrada la relación entre las arquitecturas identificadas en los laboratorios remotos desplegados para la experimentación con sistemas embebidos en el capítulo anterior y las principales características que definen el rendimiento de un laboratorio remoto desarrollado para permitir la experimentación con tecnologías embebidas, en este capítulo se definirán los componentes necesarios para el despliegue de un laboratorio remoto, se describirán las arquitecturas identificadas y se diseñará una nueva arquitectura optimizada para la consecución de las características remarcadas.

3.1 Identificación de los componentes básicos de un laboratorio remoto para la experimentación con sistemas embebidos

Tras el estudio de los principales laboratorios remotos dedicados a la experimentación con sistemas embebidos llevado a cabo en el capítulo anterior se han identificado los componentes fundamentales requeridos que deben ser integrados en la arquitectura (Figura 3.1).

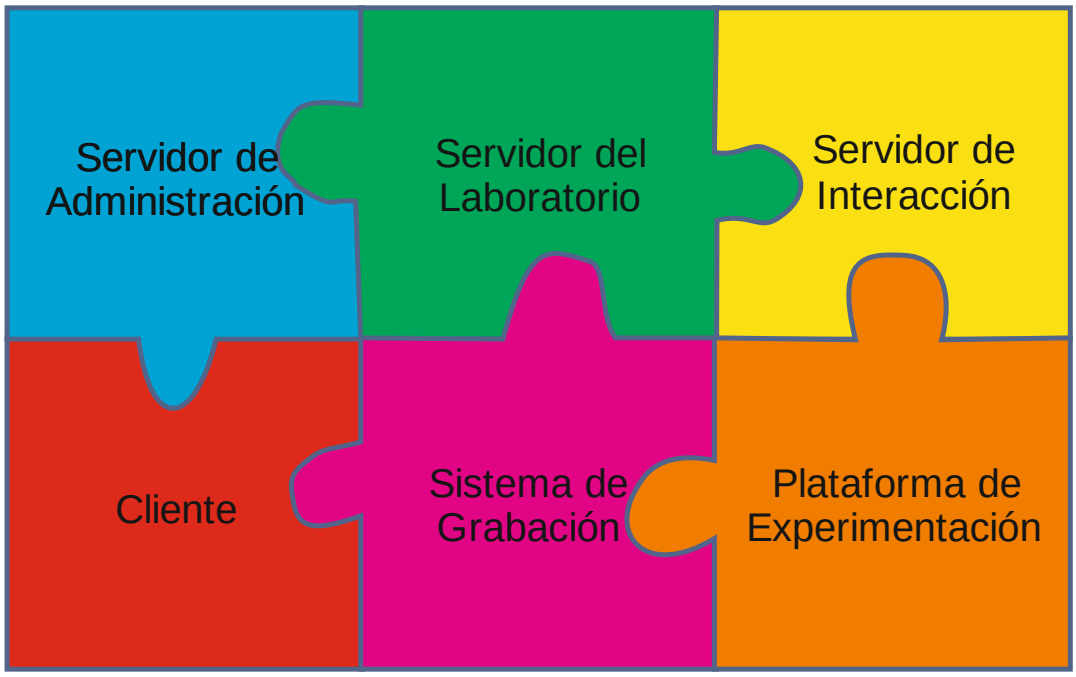

Figura 3.1. Componentes fundamentales de un laboratorio remoto para la experimentación con sistemas embebidos.

3.1.1 Servidor de administración

Este componente desempeña el conjunto de tareas de administración del laboratorio remoto. Aunque el conjunto de tareas que debe realizar un laboratorio remoto depende de los requisitos específicos del mismo, las principales tareas de administración son las siguientes:

- *Autenticación/gestión de usuarios*. La autenticación de los usuarios puede llevarse a cabo sobre un sistema específico propio del laboratorio remoto, sobre un dominio institucional o mediante un sistema de autenticación y autorización en Internet (OpenID, OAuth, FB Login, etc.).
- *Administración de permisos*. En ocasiones los laboratorios remotos permiten identificar entre tipos de usuarios con diferentes prioridades o tiempos de acceso.
- *Gestión de colas*. Son los sistemas de gestión de acceso más convenientes cuando el tiempo de la sesión es reducido. Gestionan los accesos de los usuarios concurrentes por orden de llegada o prioridad.
- *Gestión de reservas*. Permiten la reserva de laboratorio por los estudiantes durante fracciones de tiempo. En aquellos laboratorios remotos en los que la

experimentación requiere tiempos considerables este sistema es recomendable para garantizar el acceso ordenado de los estudiantes.

- *Registro de trazabilidad de las sesiones de experimentación*. Registrar las diferentes acciones que los estudiantes desarrollan en sus sesiones de experimentación, así como almacenar las diferentes versiones de los diseños implementados proporciona una información que permite a los administradores detectar problemas en la implementación y a los profesores evaluar la experimentación llevada a cabo por los estudiantes.
- *Balanceo de carga entre diferentes instancias de experimentación*. Aquellos laboratorios que disponen de múltiples instancias o de diferentes tecnologías en las plataformas de experimentación requieren de un sistema que permita balancear entre ellas las sesiones de los estudiantes (Figura 3.2).

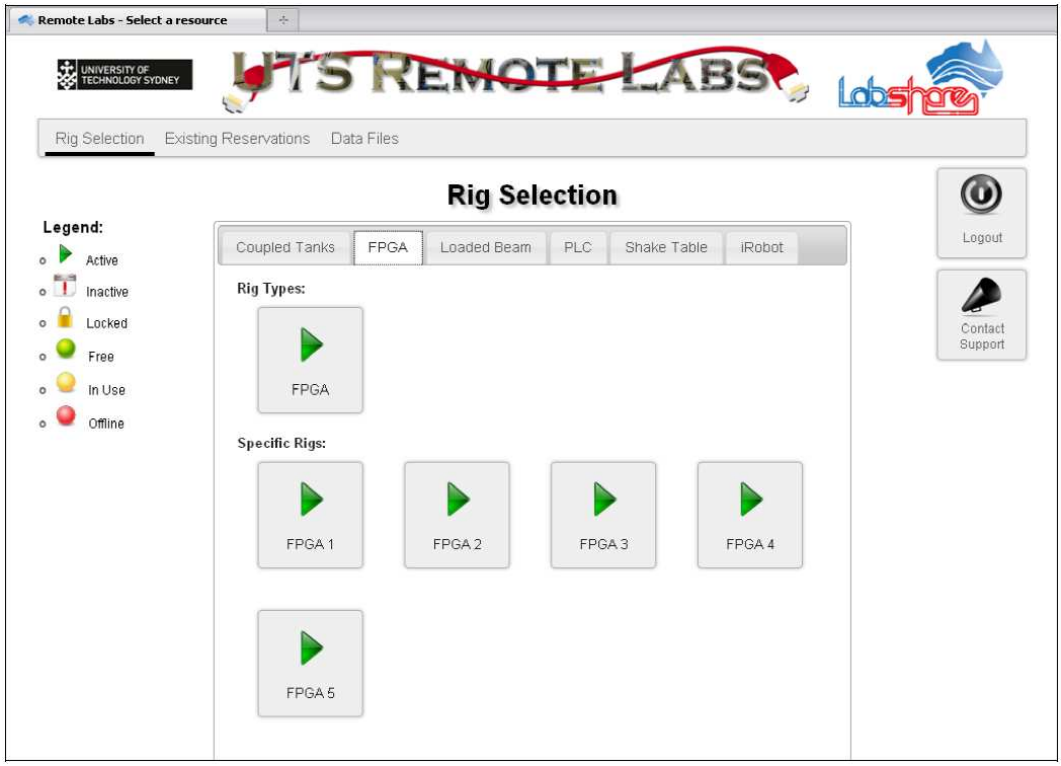

Figura 3.2. Interfaz de usuario para la selección de instancia de experimentación en el laboratorio remoto Labshare FPGA rig.

En el caso específico de los laboratorios remotos centrados en la experimentación con sistemas embebidos en los que, como se ha demostrado en el capítulo anterior, la experimentación concurrente de varios estudiantes no tiene sentido, la función principal del servidor de administración es gestionar de forma ordenada las sesiones de experimentación de los estudiantes. Esta gestión debe llevarse a cabo en base a un sistema de colas, con o sin distintas prioridades, para el dominio de estudiantes o

mediante un sistema de reservas. El sistema gestor debe elegirse en función del tiempo máximo de la sesión y de la demanda del experimento (Orduña, y otros, 2011).

Aunque los sistemas de gestión de laboratorios remoto aportan soluciones óptimas para el desarrollo de todas estas tareas, existen diseños e implementaciones "standalone" optimizados para su despliegue en dispositivos de bajo rendimiento o que facilitan su instalación para laboratorios remotos que deben ser accedidos o consumidos de forma independiente.

3.1.2 Servidor del experimento

Este componente se encarga de recibir las órdenes que los estudiantes que acceden al laboratorio ejecutan desde el cliente y encaminarlas hacia las plataformas de experimentación o interacción correspondientes.

Las principales acciones que debe ejecutar o lanzar el servidor del experimento son las siguientes:

- *Programación de la plataforma de experimentación*. En cualquier tecnología de sistemas embebidos la experimentación con un determinado diseño requiere la programación del dispositivo sobre el que se fundamenta el experimento (MCU, FPGA, DSP, etc.). En una arquitectura centralizada el servidor de grabación está directamente conectado al servidor del experimento (USB, RS232, etc…), mientras que una arquitectura distribuida promueve la existencia de un sistema independiente capaz de recibir el firmware (Ethernet, wifi, CAN, etc.) y llevar a cabo la grabación de la plataforma de experimentación.
- *Interacción con las entradas conectadas a la plataforma de experimentación*. La validación de un diseño implica la interacción con las entradas del sistema mediante sistemas de conmutación (pulsadores, interruptores, etc…), generadores de frecuencia, selectores analógicos, etc.
- *Protección sobre errores voluntarios o fortuitos*. Algunos laboratorios remotos disponen de mecanismos que permiten la detección en los diseños implementados de condiciones que pueden conllevar un uso inadecuado del laboratorio.
- *Sistemas autotesting para la detección de errores*. Es conveniente que los laboratorios remotos ejecuten periódicamente y de manera automática un conjunto de tests de validación que permitan detectar el funcionamiento erróneo de los principales componentes del laboratorio y generen las alarmas correspondientes sobre el personal de administración.

La implementación de este componente depende directamente de la arquitectura del laboratorio remoto. La mayoría de los laboratorios analizados en el capítulo anterior han desarrollado el servidor del experimento atendiendo a las características específicas de las plataformas de experimentación e interacción utilizadas, e impidiendo su réplica sobre otras tecnologías (Figura 3.3).

Los sistemas de gestión de laboratorios remotos incluyen mecanismos para la interpretación de las órdenes recibidas de los estudiantes y permiten la implementación del código requerido para su ejecución sobre las plataformas de experimentación e interacción. La descomposición de todos los componentes identificados en el presente capítulo permite desacoplar las funcionalidades de cada uno de ellos, simplificando enormemente la adecuación a diferentes tecnologías de sistemas embebidos. Así mismo la adopción de una arquitectura distribuida simplifica el desarrollo de este componente permitiendo coordinar las tareas de experimentación con independencia del sistema de programación de la plataforma de experimentación y del protocolo de comunicación utilizado con la plataforma de experimentación.

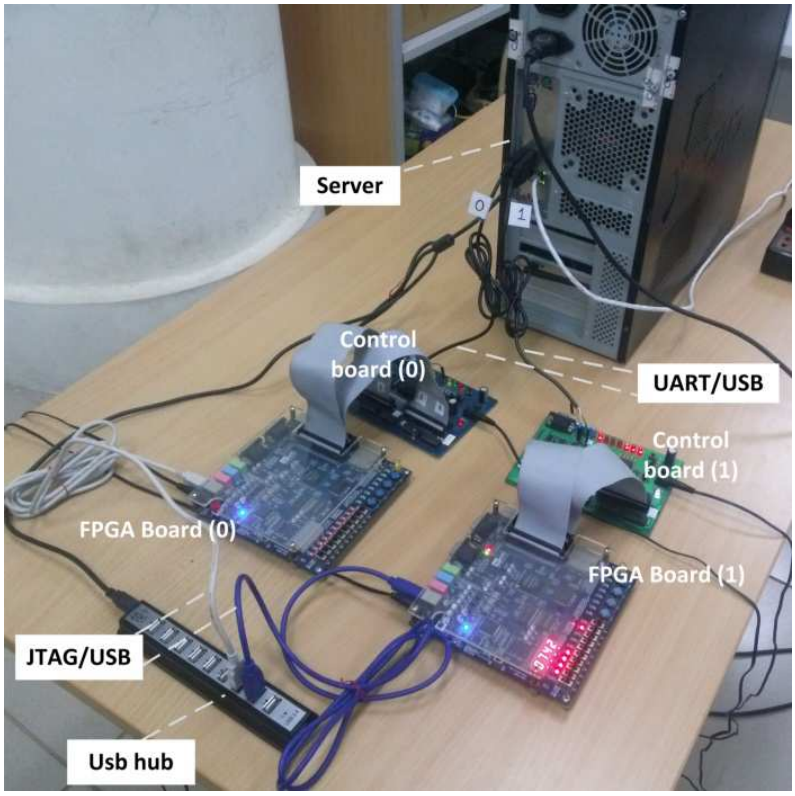

Figura 3.3. Imagen del servidor del laboratorio conectado a dos instancias de experimentación del laboratorio remoto de la Universidad de Ciencia y Tecnologías de Hanoi.

3.1.3 Cliente

Proporciona al estudiante el entorno de experimentación. La composición de este componente depende directamente de la naturaleza del laboratorio. La mayoría de los laboratorios analizados en el estado del arte se limitan a proporcionar un recurso en el cual el estudiante puede llevar a cabo las fases de programación y validación del diseño, exigiendo el desarrollo de las fases de codificación y compilación/síntesis en un entorno de desarrollo compatible con la tecnología de experimentación ajeno al laboratorio remoto. Algunas excepciones (Fotopoulos, Anastasios, & Anastasios, 2013) permiten desarrollar todas las fases de la experimentación desde el propio laboratorio remoto. Las ventajas de incorporar los recursos de edición del código y compilación/síntesis en el laboratorio remoto son obvias: permite llevar a cabo todas las fases de la experimentación sin la necesidad de instalar aplicaciones en el dispositivo desde el que se accede. Además, si el cliente ha sido desarrollado con tecnologías web estándar se universaliza el acceso al laboratorio permitiendo la experimentación desde dispositivos móviles (Orduña P. , García-Zubia, Irurzun, & Rodriguez-Gil, 2011). Por el contrario, incorporar un sistema de desarrollo en el propio laboratorio presenta también ciertas desventajas: aunque el código generado por el estudiante sea idéntico, el desarrollo de la experimentación remota se lleva a cabo mediante un sistema de desarrollo diferente al empleado en la experimentación presencial, aportando nuevas diferencias en la percepción del estudiante entre ambas experimentaciones. Paralelamente la disponibilidad universal de los laboratorios remotos puede generar problemas de licencia cuando el ensamblador, compilador o herramienta de síntesis se ejecuta desde el propio laboratorio. Por otro lado, el tiempo requerido para las sesiones del laboratorio aumenta exponencialmente al incluir las fases del diseño que más tiempo consumen en el desarrollo de un sistema embebido.

Centrando el cliente en el desarrollo de las fases de programación y validación del diseño, las tareas que deben ser implementadas en el cliente son las siguientes:

- *Selección y envío del firmware*. El cliente debe proporcionar el recurso que permita la búsqueda del archivo binario implementado en el sistema de ficheros del dispositivo cliente y su transmisión hasta el servidor del experimento.

- *Panel de control de interacción*. Generalmente la validación del diseño implementado exige la interacción del usuario mediante la manipulación de interruptores, pulsadores, potenciómetros, generador de frecuencias, sistemas sensores, etc… El cliente deber incluir un interfaz gráfico que proporcione un panel de control que permita manipular los dispositivos virtuales de interacción incluidos.

- *Panel de monitorización*. La validación del sistema requiere la monitorización del comportamiento de la plataforma de experimentación una vez ha sido programada con el diseño del estudiante. Como se ha observado en el estado del arte el recurso de monitorización más habitual en los laboratorios remotos es una cámara que proporciona el video capturado sobre la propia plataforma de experimentación y los periféricos conectados a esta. Otros laboratorios remotos monitorizan el comportamiento mediante paneles virtuales que representan el comportamiento real de la plataforma de experimentación (Morgan, y otros, 2014). Aunque la realidad virtual aporta nuevas metodologías didácticas, aumenta la separación entre la experimentación real y la remota (Figura 3.4) (Butime, y otros, 2012).

- *Información de sesión*. El cliente debe proporcionar al estudiante información sobre el desarrollo de la sesión de experimentación aportando el tiempo de experimentación restante, manual de usuario, etc.

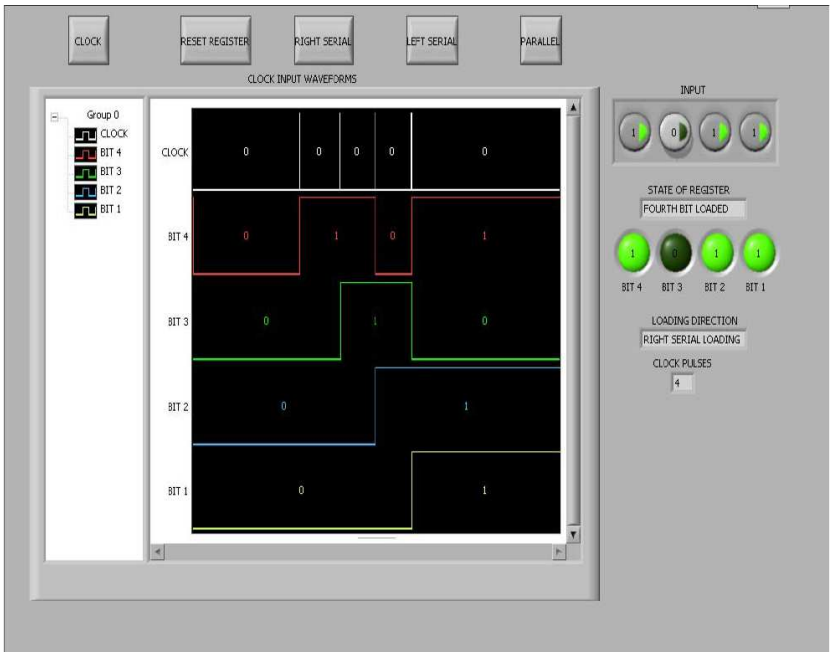

Figura 3.4. Cliente del laboratorio remoto para experimentación con FPGA desarrollado por la Universidad de Makarere (Uganda).

3.1.4 Servidor de grabación

Un requisito de todo laboratorio remoto para la experimentación sobre tecnologías de sistemas embebidos es la provisión de experimentación real. Esta premisa en la que debe fundamentarse todo laboratorio remoto exige la programación real del dispositivo fundamental de la plataforma de experimentación con el diseño

implementado por el estudiante. Dependiendo de la tecnología de experimentación, el servidor de grabación y los dispositivos programadores varían considerablemente.

En la experimentación presencial la programación del dispositivo se desarrolla generalmente mediante un dispositivo programador que se encuentra disponible en el laboratorio. Existen sistemas de desarrollo didácticos que integran el dispositivo de programación (Digilent, 2013) mientras que otros exigen el empleo de un dispositivo externo. En ambos casos el dispositivo programador debe conectarse directamente (USB, RS232, paralelo, etc…) al computador desde el que se ejecuta el sistema de desarrollo (Figura 3.5). Muchos de los laboratorios analizados en el estado del arte utilizan el mismo sistema de programación que se utiliza en la experimentación presencial, conectado directamente sobre el servidor del experimento (Hercog, Gergic, Uran, & Jezernik, 2007) (Soares & Lobo, 2011) (Kutlu A. , 2004) (Pop, Zutin, Auer, Henke, & Wuttke, 2011). Esta solución, evita el desarrollo de un servidor de grabación a medida del laboratorio remoto, pero dificulta enormemente dos de las características identificadas como clave en el análisis de los laboratorios remotos para la experimentación sobre sistemas embebidos: Escalabilidad y Replicabilidad. Por el contrario, otros laboratorios (Morgan, y otros, 2014) (Kutlu & Taşdelen, 2010) contemplan la implementación de un componente independiente que desarrolla esta tarea, recibiendo mediante un protocolo de comunicación estándar el fichero binario desde el servidor del laboratorio. Esta solución complica el desarrollo del laboratorio remoto exigiendo un mayor conocimiento de la tecnología sobre la que se desarrolla el experimento pero permite la adopción de una arquitectura distribuida que facilita la inclusión de nuevas instancias basadas en la misma u otra plataforma de experimentación, y no compromete la sostenibilidad del laboratorio que permite la experimentación sobre tecnologías con un ciclo de vida reducido.

Figura 3.5. Imagen del servidor de grabación ARM-USB-OCD de Olimex para la programación de microcontroladores ARM por medio de RS232 o USB.

Como alternativa a los sistemas de programación convencionales algunos laboratorios remotos se aprovechan de la capacidad de reconfiguración que disponen algunos de los dispositivos sobre los que se fundamenta la experimentación. Aunque los dispositivos programables también disponen de esta capacidad de reconfiguración (Sánchez-Román, y otros, 2010) (Gabriel & Ovidiu, 2003), es en los microcontroladores donde más se emplea esta técnica para la programación del dispositivo (Limpraptono, Ratna, & Sudibyo, 2012) (Fotopoulos, Anastasios, & Anastasios, 2013). La capacidad de la mayoría de los microcontroladores de sobre escribir en tiempo de ejecución su propia memoria de programa permite el desarrollo de pequeños programas cuya funcionalidad es precisamente recibir mediante un bus de comunicaciones estándar (UART, ETHERNET, I2C, CAN, etc..) el código máquina que deben ejecutar y realojarlo convenientemente en la memoria de programa para al finalizar la transferencia del mismo lanzar su ejecución. Esto programas, denominados bootloaders, facilitan enormemente la programación de los dispositivos sobre los que se desarrolla la experimentación y abaratan el coste del laboratorio pero exigen configuraciones específicas que el estudiante debe tener en cuenta durante el desarrollo del firmware que difieren de las condiciones reales y provocan la ejecución de acciones diferentes a las desarrolladas en la experimentación real. En ocasiones, estas configuraciones que suelen consistir fundamentalmente en la alteración de los vectores de reset e interrupciones, pueden corregirse mediante el desarrollo de un fichero de linkaje a medida o mediante la configuración del sistema de desarrollo, minimizando estas diferencias. En el apartado 4.1.1.1 del capítulo 4 se explicará el desarrollo de un bootloader que cumple con estas características. Otra alternativa es la adecuación del firmware desde el servidor del laboratorio. Existen laboratorios remotos (García-Zubia, Angulo, Hernández-Jayo, & Orduña, 2008), en los que el servidor del laboratorio no actúa simplemente como intermediario entre el cliente y la plataforma de experimentación, sino que además lleva a cabo un análisis del firmware provisto por el estudiante adecuando las posiciones de memoria del propio firmware a los requisitos del laboratorio y a la vez protegiendo el código del bootloader ante una sobre escritura llevada a cabo por el estudiante de manera fortuita o intencionada. De esta forma el empleo del bootloader es completamente transparente para el usuario.

3.1.5 Servidor de interacción

La funcionalidad de este componente es generar las señales físicas que son provistas a la plataforma de experimentación en respuesta a las órdenes enviadas desde el panel de interacción incluido en el cliente.

La implementación del servidor de experimentación requiere de un componente hardware para generar físicamente las señales correspondientes a la interacción del estudiante. Aunque es fundamental proporcionar una experimentación con la máxima semejanza a la que el estudiante desempeña en un laboratorio real, la comunicación entre el cliente y el servidor del experimento, su posterior envío desde éste al servidor de interacción y la propia generación de la señal provocan un retraso que debe minimizarse.

El estado del arte llevado a cabo de los laboratorios presenta diversas implementaciones del servidor de interacción que se resumen a continuación.

- Algunos laboratorios remotos para experimentación con dispositivos programables (Lobo, 2011) (Morgan, y otros, 2014) utilizan recursos adicionales del propio dispositivo sobre el que se programa el diseño del estudiante para generar la señales que retroalimentan la implementación del estudiante. Aunque la adecuación de los niveles lógicos está garantizada, obviamente esta solución condiciona totalmente la tecnología y no es replicable a otras plataformas de experimentación. Además en estas implementaciones del servidor de interacción, las órdenes son enviadas a través del propio servidor de grabación (JTAG) que provoca alteraciones en registros del dispositivo impidiendo la ejecución de órdenes concurrentes.

- Otros laboratorios (Pop, Zutin, Auer, Henke, & Wuttke, 2011) utilizan tarjetas específicas de adquisición de datos que proporcionan los driver necesarios para su control desde entornos avanzados de desarrollo para el diseño de sistemas (p.e. LabView). Estas tarjetas facilitan enormemente el desarrollo del panel de interacción en el cliente pero encarecen el coste del laboratorio. Además, estar tarjetas están diseñadas para facilitar la experimentación presencial y exigen una conexión directa al servidor del laboratorio (PCI, USB, RS232, etc.) dificultando la escalabilidad del laboratorio.

- Finalmente muchos laboratorios (Goncalves de Moraes & Mendes de Sales, 2012) (Limpraptono, Ratna, & Sudibyo, 2012) (Reichenbach M. , Schmidt, Pfundt, & Fey, 2011) utilizan microservidores para la implementación del servidor de interacción. Los enormes avances en microcontroladores han permitido la aparición de numerosas plataformas embebidas de bajo coste (Microchip Technology, 2011) que incorporan conectividad LAN a través de Ethernet o wifi y permiten la implementación de Servidores HTTP capaces de interpretar peticiones HTTP y generar las señales físicas correspondientes a través de sus puertos de expansión. Estos sistemas garantizan la escalabilidad del laboratorio y son completamente independientes de la tecnología garantizando la replicabilidad sobre cualquier dispositivo (Figura 3.6).

Figura 3.6. Imagen del computador de factor de forma reducido BeagleBone diseñado por Texas Instruments con licencia de hardware abierto.

El empleo de diferentes tecnologías para la implementación del servidor de interacción, la plataforma de experimentación e incluso los periféricos controlados por esta exigen el empleo de sistemas de adecuación de señal que equipare los niveles lógicos utilizados entre los diferentes sistemas (Angulo, García Zubia, & Asunción, 2014).

Los laboratorios remotos deben permitir llevar a cabo una experimentación similar a la que los estudiantes desarrollan en laboratorios presenciales. Una de las limitaciones encontradas en la mayoría de los laboratorios remotos analizados en el estado del arte (Soares & Lobo, 2011) (Kutlu & Taşdelen, 2010) (Limpraptono, Ratna, & Sudibyo, 2012) (Lobo, 2011) (Morgan & Cawley, 2011) (Pop, Zutin, Auer, Henke, & Wuttke, 2011) (Fotopoulos, Anastasios, & Anastasios, 2013) es que únicamente permiten la interacción con entradas y salidas digitales. Los servidores de interacción deben permitir llevar a cabo los mismos experimentos que se llevan a cabo en la experimentación real añadiendo variables analógicas, buses de comunicación, modulación por ancho de pulsos, etc.

3.1.6 Plataforma de experimentación

Se trata del sistema de desarrollo hardware sobre el que se lleva a cabo la experimentación. En algunos laboratorios remotos (Kutlu & Taşdelen, 2010) (Morgan, y otros, 2014) la plataforma de experimentación se ha desarrollado a medida en

función de los requisitos que el desarrollador del laboratorio establece para la experimentación. Esta elección permite disponer todos los periféricos de entrada y salida con los que se desarrollará la experimentación con una interconexión fija que permite la interacción simultánea con todos ellos. En otros casos (Hercog, Gergic, Uran, & Jezernik, 2007) (Soares & Lobo, 2011) (Limpraptono, Ratna, & Sudibyo, 2012) (Pop, Zutin, Auer, Henke, & Wuttke, 2011) (Reichenbach M. , Schmidt, Pfundt, & Fey, 2011) (Fotopoulos, Anastasios, & Anastasios, 2013) la plataforma de experimentación elegida para el laboratorio remoto es la misma que se emplea en los laboratorios presenciales del centro educativo (Figura 3.7).

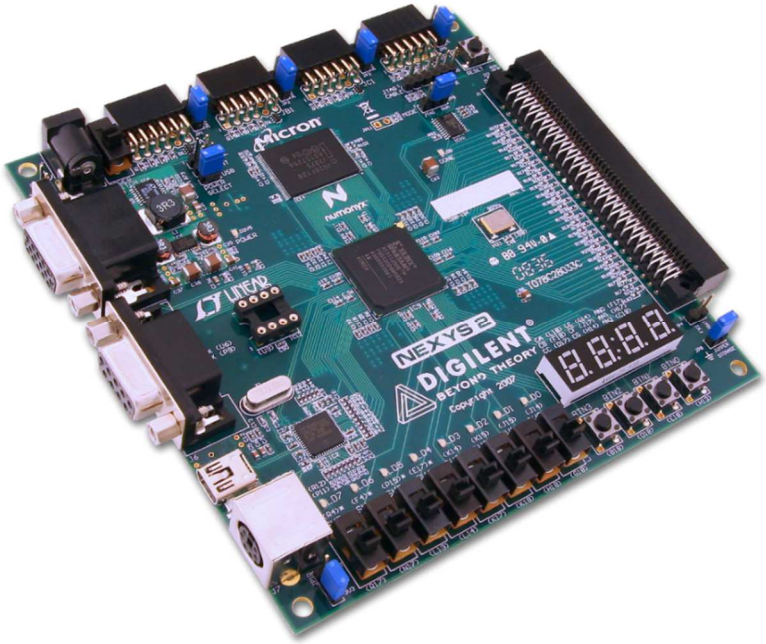

Figura 3.7. Imagen del sistema de desarrollo para la experimentación con FPGA Nexys 2. Utilizado en los laboratorios remotos de la Universidad de Erlangen-Nuremberg y de la Universidad de Galway.

Esta opción permite utilizar el laboratorio remoto para asentar los contenidos estudiados y maximizar el tiempo dedicado a la experimentación presencial. Sin embargo en multitud de ocasiones los sistemas de desarrollo diseñados para la experimentación presencial disponen de jumpers y mecanismos que permiten al estudiante seleccionar los periféricos que se conectan a los puertos del dispositivo, reduciendo las posibilidades del laboratorio remoto. Además la disposición de los periféricos de salida en ocasiones no es la más conveniente para permitir la monitorización desde una cámara.

3.1.7 Rig o instancia de experimentación

Aunque todos los componentes identificados disponen de entidad propia e intercambian unívocamente comandos independientes con el servidor del experimento, la conexión directa requerida entre el servidor de grabación y el servidor de interacción con la plataforma de experimentación impiden la separación física de estos componentes. Por este motivo, a la combinación de estos componentes la denominaremos "Instancia de Experimentación" o adoptando la nomenclatura del sistema de gestión de laboratorios Sahara de Labshare, "Rig" (Figura 3.8).

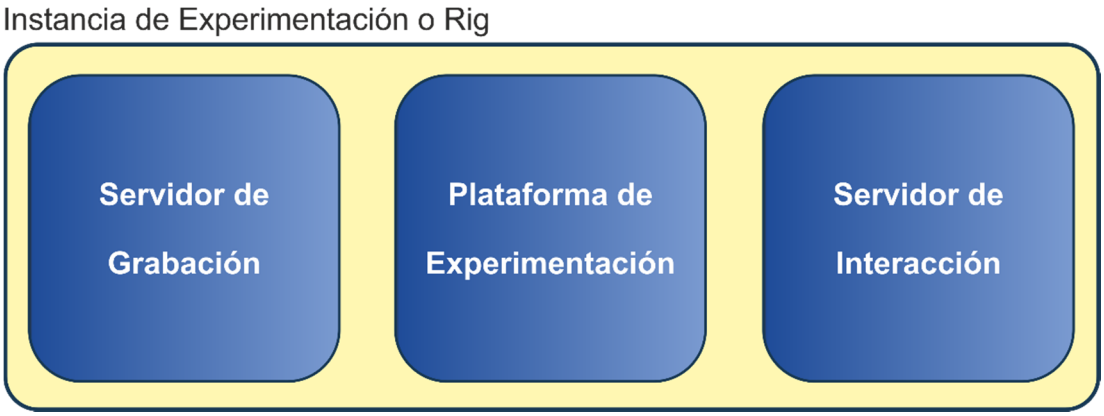

Figura 3.8. Componentes que conforman la instancia de experimentación o rig de un laboratorio remoto para experimentación en sistemas embebidos.

Como ya se ha comentado anteriormente, la naturaleza no concurrente de los laboratorios remotos para experimentación con sistemas embebidos requiere la disposición de múltiples plataformas de experimentación para soportar grandes dominios de estudiantes. Generalmente, cada plataforma de experimentación requiere de un servidor de grabación y un servidor de interacción correspondiente. Por ello la escalabilidad de los laboratorios, una de las características definidas como fundamentales en el estado del arte, requiere la disposición múltiples instancias completas de experimentación.

Por otro lado, para cumplir con la característica de replicabilidad de tal forma que el laboratorio no dependa de una tecnología específica de experimentación es fundamental mantener independientes los componentes que conforman el rig (Figura 3.9).

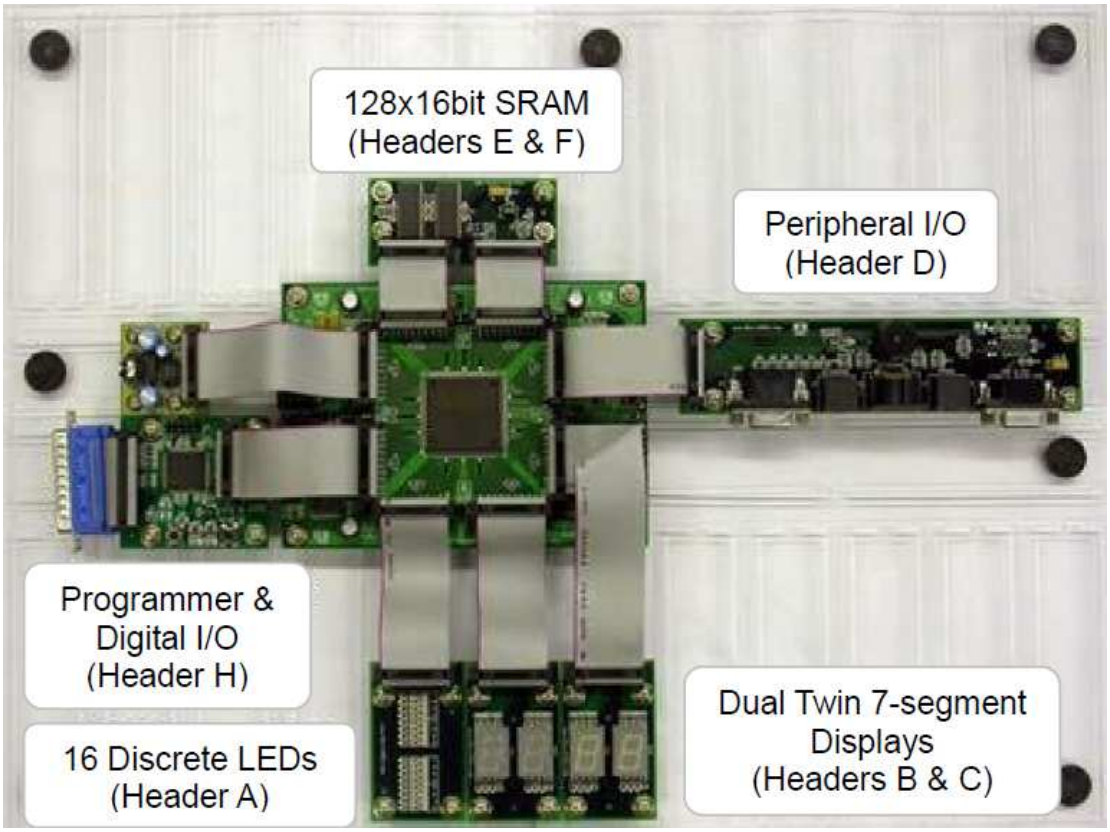

Figura 3.9. Instancia de experimentación del laboratorio remoto Labshare FPGA.

3.2 Tipología de arquitecturas de laboratorio remoto para la experimentación con sistemas embebidos

La disposición e implementación de los componentes identificados en los laboratorios analizados permite identificar cuatro arquitecturas básicas:

- *Arquitectura Centralizada*. Todos los componentes están directamente conectados sobre el servidor del laboratorio.
- *Arquitectura Distribuida*. Todos los componentes mantienen su independencia y la información se transfiere entre ellos principalmente mediante una red de comunicaciones.
- *Arquitectura Mixta*. Aunque la interconexión de los principales componentes se implementa sobre una red de comunicaciones, la arquitectura requiere una conexión directa entre algún componente de cada instancia y el servidor del laboratorio.
- *Arquitectura Replicada*. Como solución a la falta de escalabilidad de los laboratorios remotos con arquitectura centralizada, muchos desarrolladores

han optado por replicar la totalidad del laboratorio remoto, compartiendo únicamente el servidor de administración entre las múltiples instancias desplegadas del laboratorio.

3.2.1 Arquitectura centralizada

Históricamente uno de los principales errores que se han cometido en el desarrollo de un laboratorio remoto para experimentación con sistemas embebidos ha consistido centrar el diseño del mismo y la implementación de cada componente en la plataforma de experimentación. Este enfoque, que puede resultar válido para la remotización de experimentos de otra naturaleza (física, astronomía, etc…) (Rochadel, da Silva, da Silva, Luz, & Alves, 2012) (Watterson, Yeoh, Giampietro, Chapman, & Houghton, 2010) (Harward, y otros, 2008) sin caducidad temporal, ha ocasionado la obsolescencia prematura de laboratorios remotos para la experimentación con microcontroladores o dispositivos programables motivada por la discontinuación, por parte del fabricante, de familias de dispositivos embebidos y de los entornos de desarrollo asociados a las mismas. Como se ha comentado anteriormente, existe una dependencia directa entre la replicabilidad del laboratorio y su sostenibilidad.

La adopción de esta arquitectura facilita el desarrollo del laboratorio remoto ya que no requiere el diseño de dispositivos hardware para la implementación de las instancias de experimentación, puesto que suele utilizar exclusivamente recursos diseñados para el desarrollo de experimentación presencial. Sin embargo implica limitaciones que afectan fundamentalmente a la escalabilidad y la replicabilidad.

Una de las características principales que definen un laboratorio remoto con arquitectura centralizada es la conexión directa de los componentes del rig (servidor de grabación, servidor de experimentación y plataforma de experimentación) al servidor del laboratorio. La mayoría de sistemas comerciales para el desarrollo de sistemas embebidos disponen de conexión directa con el computador en el que se instala el pertinente entorno de desarrollo, mediante buses de comunicación diseñados para el control de periféricos y dispositivos electrónicos. Estos buses (USB, RS232, FireWire, Paralelo, SATA, etc.) requieren conexión directa con el computador, y, o bien no permiten conexiones múltiples, o estas son muy limitadas.

En la Figura 3.10 se puede observar un ejemplo habitual de laboratorio remoto con arquitectura centralizada. Como se puede observar, la disposición del servidor de grabación y de la plataforma de experimentación adopta la instalación definida para la experimentación presencial con las tecnologías embebidas.

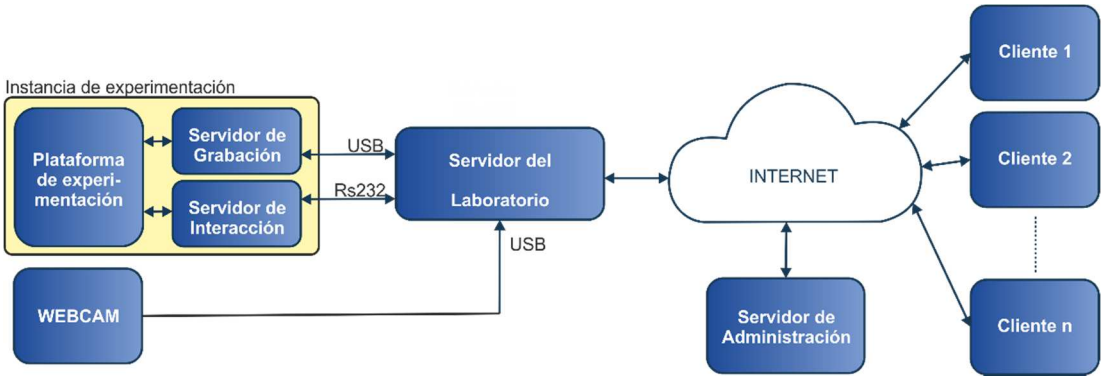

Figura 3.10. Ejemplo de arquitectura centralizada de un laboratorio remoto para la experimentación con tecnologías embebidas.

La publicación en Internet de un laboratorio para la experimentación con sistemas embebidos mediante una arquitectura centralizada se desarrolla cumpliendo con las siguientes tareas:

1. *Elección e instalación del sistema de desarrollo* para tecnologías embebidas a remotizar. Sobre el computador que conformará el servidor del laboratorio se debe conectar el sistema de desarrollo sobre el que se desarrollará la experimentación remota y el conjunto de herramientas software que permiten desarrollar la experimentación presencialmente. Como se ha dicho anteriormente, esta arquitectura centralizada se fundamenta en el sistema de desarrollo sobre el que se experimenta.

2. *Desarrollo e interconexión del servidor de interacción*. Todos los periféricos de entrada integrados en el sistema de desarrollo y que pueden ser manipulados directamente por el estudiante en la experimentación presencial deben ser sustituidos por un servidor de interacción que permita su emulación mediante comandos enviados desde el servidor del laboratorio. En la mayoría de laboratorios remotos que emplean esta arquitectura el servidor de interacción se implementa mediante un microcontrolador conectado directamente al servidor del laboratorio mediante un bus de comunicaciones integrado en el propio MCU (USB o UART).

3. *Instalación de un sistema de monitorización*. Esta fase suele consistir en añadir una cámara que apunta al sistema de desarrollo directamente conectada al servidor del laboratorio.

4. *Desarrollo del servidor del laboratorio*. Consiste en el desarrollo del software que permite la experimentación con el sistema de desarrollo sin requerir la manipulación directa sobre este. Fundamentalmente se trata de un servicio persistente o daemon (Disk and Execution Monitor) que se encarga de lanzar automáticamente la ejecución del software de grabación compatible con el sistema de desarrollo cuando recibe un firmware, así como de generar sobre

el servidor de interacción las órdenes demandadas. Aunque este componente puede ser directamente desarrollado mediante los principales entornos de desarrollo software compatibles con la plataforma compatible el sistema de desarrollo, es muy habitual que se utilicen entornos de desarrollo para el diseño sistemas, con un lenguaje de programación visual gráfico (Labview) que facilita considerablemente la implementación del componente.

5. *Desarrollo del Cliente*. Mediante este componente se desarrolla el interfaz gráfico que permite al estudiante monitorizar el sistema de desarrollo, así como enviar fácilmente al servidor de administración los diseños y órdenes de interacción. Aunque suelen utilizarse tecnologías web que facilitan el acceso del estudiante mediante un navegador, también es habitual que para estos laboratorios remotos se desarrollen aplicaciones de escritorio (Kutlu A. , 2004) o clientes autogenerados por el entorno utilizado para el diseño de sistemas en el desarrollo del servidor del laboratorio. El consumo de la cámara que permite monitorizar el resultado de la experimentación puede hacerse mediante un sistema de streaming que permite visualizar en una parte del interfaz gráfico el video capturado por la cámara.
6. *Desarrollo del servidor de administración*. La mayoría de servidores que eligen esta arquitectura suelen optar por el desarrollo a medida de este componente implementando un sistema de autenticación basado en un dominio propio de usuarios y un sistema de colas o reservas para administrar las sesiones de los estudiantes (Fotopoulos, Anastasios, & Anastasios, 2013) (Kutlu & Taşdelen, 2010). En ocasiones la integración del laboratorio en un sistema de gestión de laboratorios remotos garantiza el correcto consumo del sistema (Orduña, 2013) (Pop, Zutin, Auer, Henke, & Wuttke, 2011).

La facilidad y rapidez que permite la publicación de laboratorios remotos mediante esta arquitectura genera dos problemas fundamentales que condicionan el futuro consumo del mismo por los estudiantes:

- Los buses de comunicación utilizados para conectar el servidor de grabación y el servidor impiden la escalabilidad del laboratorio reduciendo el número de instancias de experimentación a una o unas pocas dificultando el consumo del laboratorio por grandes dominios de estudiantes.
- Todos los componentes del laboratorio remoto han sido desarrollados a la medida del sistema de desarrollo escogido. La discontinuidad del mismo y su reemplazo por una nueva tecnología requiere el desarrollo de un nuevo laboratorio desde el principio. Por tanto, la carencia de replicabilidad de estos laboratorios compromete la sostenibilidad de los mismos.

3.2.2 Arquitecturas distribuidas

En contraposición con los sistemas que adoptan la arquitectura centralizada que se basan en los sistemas de desarrollo utilizados en los laboratorios presenciales, los laboratorios remotos para la experimentación con sistemas embebidos con arquitectura distribuida parten del concepto de experimentación deseada y requieren el desarrollo de los componentes individuales requeridos para permitir su desarrollo a través de Internet.

La implementación de estos laboratorios requiere mayor tiempo de desarrollo y generalmente conlleva el diseño de parte de los componentes hardware que componen la instancia de experimentación (Morgan & Cawley, 2011) (Reichenbach M. , Schmidt, Pfundt, & Fey, 2011).

La característica principal que define un laboratorio remoto que cumple con esta arquitectura consiste en que las instancias de experimentación son consideradas como elementos independientes del laboratorio. La forma de conectar las instancias de experimentación con el servidor del laboratorio condiciona tanto la escalabilidad del sistema como el rendimiento del mismo (Figura 3.11).

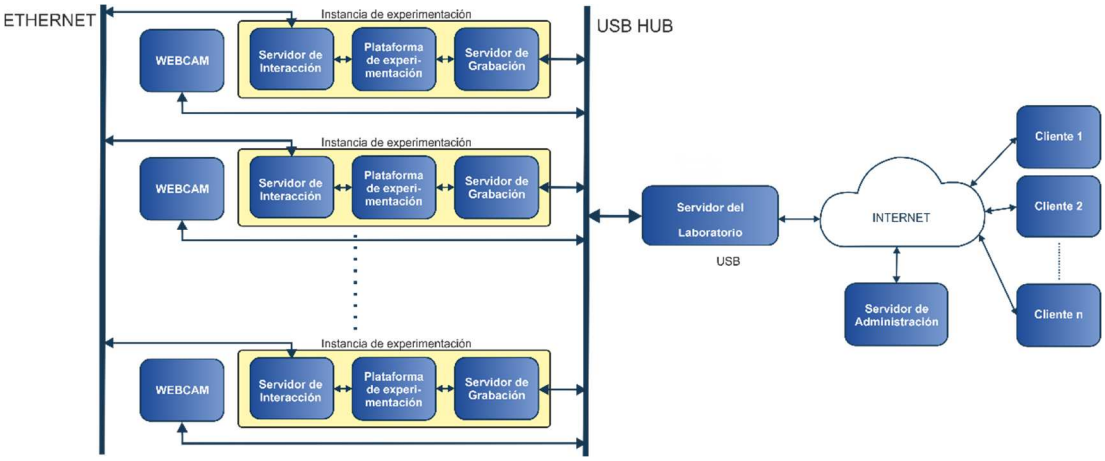

Figura 3.11. Ejemplo de arquitectura distribuida utilizada en laboratorios remotos para la experimentación con tecnologías embebidas.

Como se ha comentado en el apartado de arquitecturas centralizadas la mayoría de sistemas de desarrollo para dispositivos embebidos utiliza herramientas de programación que se conectan directamente sobre el computador que ejecuta el entorno de desarrollo software. La adecuación de los sistemas de desarrollo comerciales con este tipo de arquitectura puede exigir la inclusión de un computador dedicado a la programación del dispositivo embebido por cada instancia de experimentación. Esta solución facilita el desarrollo de laboratorios remotos acordes

con arquitectura distribuida pero eleva el coste del sistema (Lowe, Murray, Lindsay, & Liu, 2009) (Hoang, y otros, 2015).

La alternativa a los sistemas de grabación comerciales conectados directamente a un computador consiste en el desarrollo de un servidor de grabación a medida del dispositivo basado en una plataforma embebida con conectividad a redes de comunicaciones. Este sistema exige al desarrollador del laboratorio remoto analizar la tecnología de programación y diseñar un sistema compatible con la misma capaz de recibir el archivo binario a programar en el dispositivo y de ejecutar dicha programación (Henke, Tabunshchyk, Wuttke, Vietzke, & Ostendorff, 2014). En ocasiones, el correcto empleo de bootloaders o la existencia de dispositivos comerciales de programación con conectividad Ethernet facilitan la implementación de este componente del laboratorio remoto (Morgan, Cawley, & Newell, 2012).

Respecto al servidor de interacción, la proliferación de plataformas embebidas de factor forma reducido ("small form factor computer") (Krushinitskiy & Sziebig, 2013) permite disponer de plataformas conectables de bajo coste con los puertos de conexión suficientes para la generación de la señales físicas que los estudiantes demandan desde el cliente. La consideración independiente del servidor de interacción respecto a la plataforma de experimentación suele implicar el desarrollo de sistemas para la adecuación de los niveles entre ambos componentes.

La implementación independiente de las instancias de experimentación utilizada por los laboratorios remotos con arquitectura distribuida proporciona las siguientes ventajas:

- *Escalabilidad*. Naturalmente los laboratorios remotos con arquitectura distribuida permiten el despliegue de un número significativo de instancias, siendo la única limitación la impuesta por el protocolo de comunicaciones utilizado para conectar las instancias con el servidor del laboratorio.
- *Desplegabilidad*. La entidad independiente de las instancias de experimentación facilita el despliegue de los laboratorios remotos y su disposición en las instalaciones sobre las que se despliegan. Si bien es necesario disponer la documentación pertinente a todos los sistemas hardware desarrollados para el laboratorio (Powell, 2012).

Sin embargo, es común en laboratorios con esta arquitectura que la dependencia con las plataformas de experimentación y en concreto con los sistemas de grabación utilizados, complique la replicabilidad del laboratorio a otras tecnologías, lo que constituye una desventaja evidente para esta arquitectura.

Generalmente, los laboratorios remotos basados en arquitecturas distribuidas son integrados en sistemas de gestión de laboratorios remotos (RLMS) (Lowe,

Machet, & Kostulski, 2012) o sistemas de gestión del aprendizaje (LMS) (Castro, y otros, 2010) (San Cristóbal, 2010) (García-Zubia, Orduña, Irurzun, Angulo, & Hernandez-Jayo, 2009) (San Cristobal, y otros, 2011) que facilitan el desarrollo del servidor de administración proporcionando recursos como la gestión de colas y reservas con soporte para el balanceo de carga entre diferentes instancias de experimentación (Orduña, y otros, 2014).

Finalmente la independencia de los componentes que conforman las instancias de experimentación facilita enormemente el desarrollo del servidor del laboratorio, limitando su actividad a la de un mero enrutador entre los comandos recibidos desde el cliente por parte de los estudiantes hacia los componentes de la instancia correspondiente con la que han establecido una sesión de experimentación.

3.2.3 Arquitecturas mixtas

Algunos laboratorios remotos (Kutlu & Taşdelen, 2010) (Henke, Ostendorff, Wuttke, & Vogel, 2012) (Hercog, Gergic, Uran, & Jezernik, 2007) (Limpraptono, Ratna, & Sudibyo, 2012) comparten características de ambas arquitecturas. Estos laboratorios, que son incluidos en una tercera tipología denominada mixta, procuran la interconexión independiente de los componentes que los conforman, pero presentan alguna excepción que impide englobarlos en la categoría de laboratorio remoto con arquitectura distribuida (Figura 3.12).

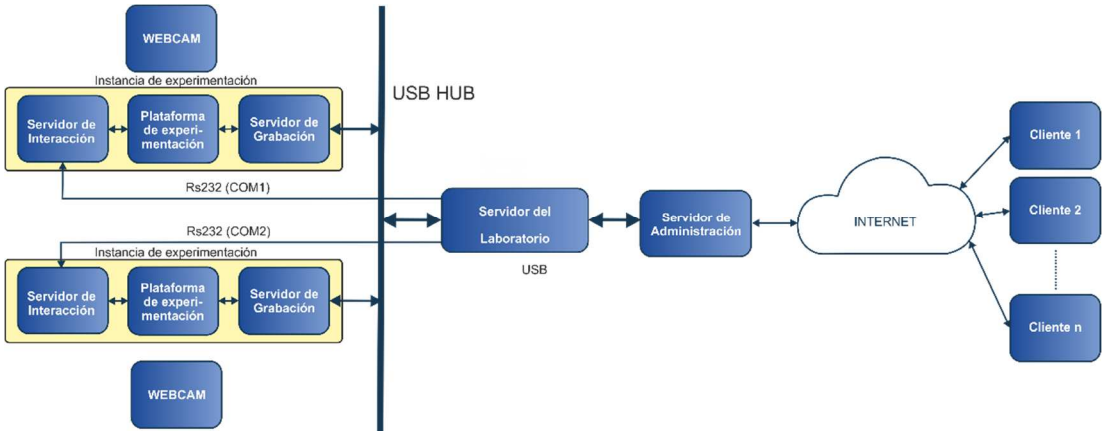

Figura 3.12. Ejemplo de arquitectura mixta donde la conexión directa con los servidores de interacción condiciona la escalabilidad del laboratorio remoto.

El control de estos componentes directamente desde el servidor del laboratorio se debe a diferentes motivaciones. Ciertos laboratorios (Hercog, Gergic, Uran, & Jezernik, 2007) (Limpraptono, Ratna, & Sudibyo, 2012) disponen todos sus componentes de forma distribuida mediante una red de comunicaciones, pero

realizan la programación de los dispositivos embebidos directamente desde el servidor del laboratorio. Esta solución permite una escalabilidad moderada, aceptando un número de instancias reducido en base al bus de comunicaciones utilizado. Además limita la complejidad de la actualización a nuevas plataformas de experimentación limitando esta al cambio de los sistemas de grabación y al software utilizado para esta tarea. Sin embargo, estos laboratorios no permiten concurrencia de diferentes tecnologías y su despliegue es complicado.

En otras ocasiones (Kutlu & Taşdelen, 2010) es el propio servidor de interacción el que condiciona esta arquitectura mixta, permitiendo fácilmente replicar el laboratorio hacia nuevas tecnologías pero limitando la escalabilidad del mismo en base a un bus de comunicaciones que controla la interacción con las plataformas de experimentación.

Finalmente hay laboratorios (Henke, Tabunshchyk, Wuttke, Vietzke, & Ostendorff, 2014) que persiguen el control de dispositivos desde múltiples plataformas de experimentación condicionando la conectividad entre dichos dispositivos y las plataformas por medio de un Bus desarrollado a la medida del laboratorio, lo que condiciona la interconexión de los componentes. En este caso tanto la replicabilidad como la escalabilidad están garantizadas, y es la naturaleza de la experimentación la que se ve limitada a la capacidad del transporte físico de las señales a través del Bus del laboratorio (Henke, Vietzke, Wuttke, & Ostendorff, 2015).

3.2.4 Arquitecturas replicadas

Este último tipo de arquitectura identificado surge de la necesidad de escalar laboratorios con arquitectura centralizada. Como se ha explicado anteriormente, la arquitectura centralizada impide la escalabilidad de los laboratorios, impidiendo su consumo por dominios importantes de estudiantes. Una solución bastante difundida para permitir la publicación de múltiples instancias en laboratorios con arquitecturas centralizadas consiste en la replicación de conjunto total del laboratorio (Hercog, Gergic, Uran, & Jezernik, 2007) (El-Medani, 2008) (Sell, y otros, 2015) (Vandorpe, y otros, 2013)(Figura 3.13).

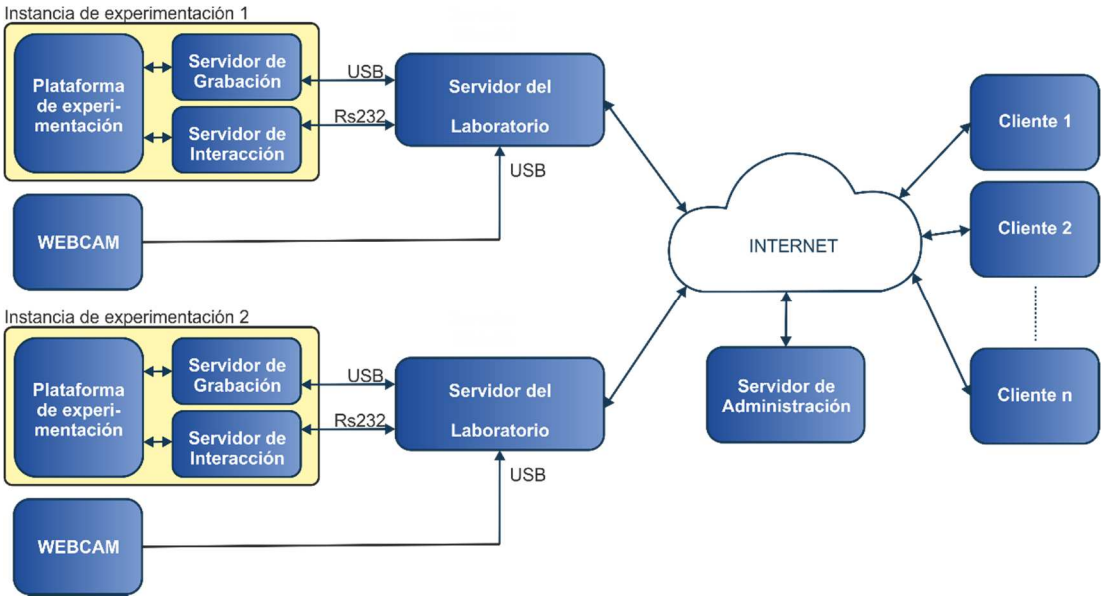

Figura 3.13. Ejemplo de laboratorio remoto con arquitectura replicada.

La replicación de los laboratorios implica el diseño del componente del servidor de administración que permita el balanceo de carga entre los diferentes servidores de laboratorio que controlan cada instancia independiente (Figura 3.14). La utilización de sistemas de gestión de laboratorios remotos, que suelen soportar nativamente esta característica, simplifica enormemente la implementación de este tipo de sistemas, garantizando una escalabilidad progresiva.

De este modo, la arquitectura replicada permite escalar laboratorios remotos basados en arquitectura centralizada de una forma muy sencilla pero que lógicamente supone un enorme encarecimiento del sistema ya que como se ha indicado anteriormente en los laboratorios remotos para la experimentación con sistemas embebidos, son los componentes externos a la propia plataforma de experimentación los que suponen el principal coste del mismo.

Además, esta arquitectura solventa la carencia de escalabilidad de la arquitectura centralizada, pero hereda el resto de carencias de esta arquitectura puesto que la dependencia con la plataforma de experimentación impide la replicabilidad del sistema hacia otras tecnologías embebidas.

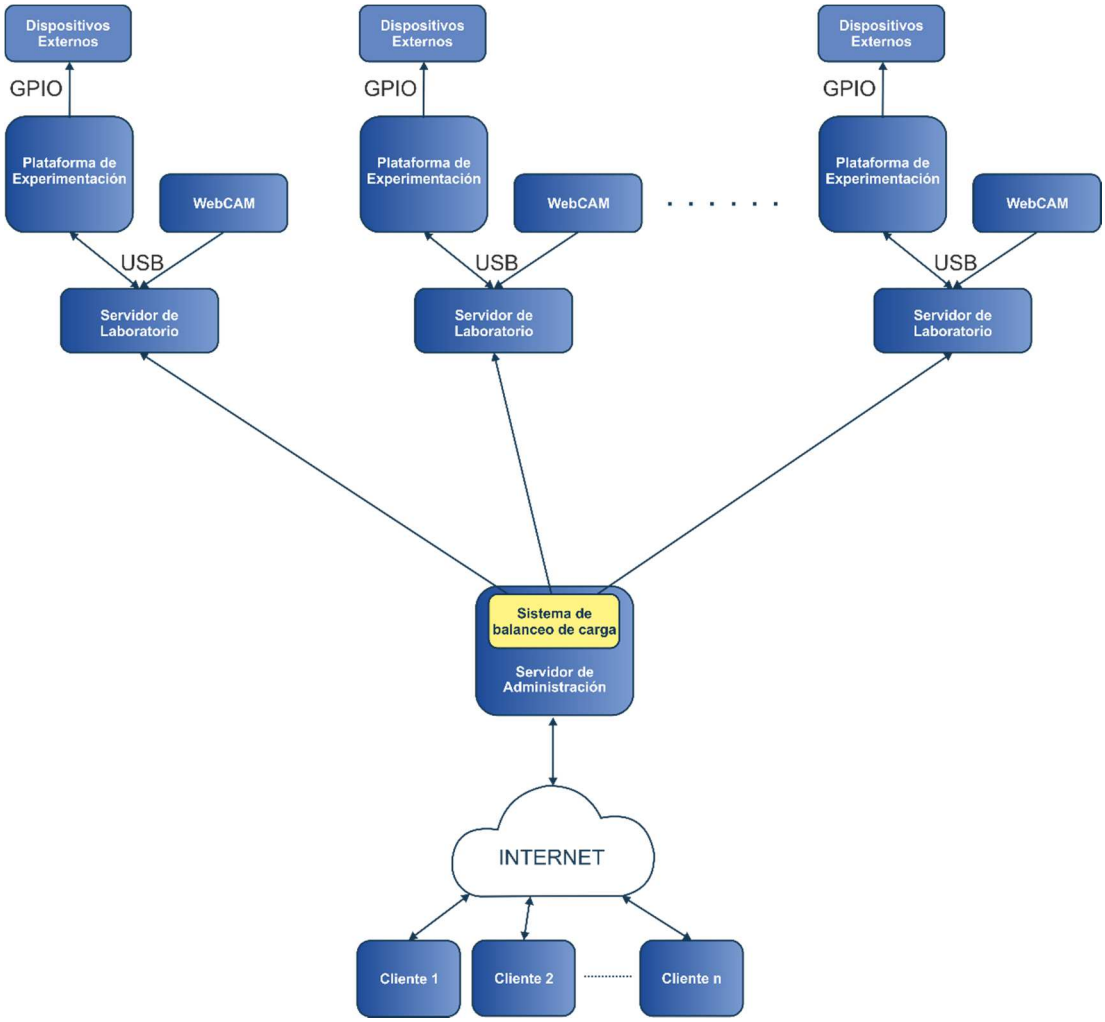

Figura 3.14. Ejemplo de arquitectura replicada para el despliegue de laboratorios remotos.

3.3 Propuesta de arquitectura abierta de fácil despliegue para permitir la experimentación con sistemas embebidos desde laboratorios remotos

A diferencia de las arquitecturas utilizadas en los laboratorios remotos analizados en el estado del arte y definidas en este capítulo, la arquitectura propuesta en esta tesis no está centrada en la tecnología de experimentación ni en los experimentos específicos a soportar. El principal objetivo de esta arquitectura es facilitar el despliegue de laboratorios remotos para permitir la experimentación con las diferentes tecnologías referentes a sistemas embebidos.

Identificadas las etapas comunes al diseño de sistemas embebidos y los componentes necesarios para el desarrollo de estas tareas desde un laboratorio remoto, la arquitectura propuesta se fundamenta en la búsqueda de la topología

óptima que permita implementar laboratorios remotos de acuerdo a las principales características demandados por estos sistemas cuando su objetivo es la experimentación con sistemas embebidos:

- ***Escalabilidad*** → Es necesario que el laboratorio remoto permita un número de instancias de experimentación que se adapte al número de estudiantes con acceso.
- ***Replicabilidad*** → Es fundamental permitir la renovación periódica de la tecnología sobre la que se lleva la experimentación para mantener la funcionalidad del sistema de acuerdo al ciclo de vida de los dispositivos embebidos. Además es fundamental permitir no solo sustituir la plataforma de experimentación sino la adecuación de los experimentos a la nueva tecnología.
- ***Desplegabilidad*** → El despliegue de laboratorios remotos para la experimentación con sistemas embebidos debe ser accesible a los profesionales de esta tecnología incluso careciendo de conocimientos o experiencia con la experimentación remota.

3.3.1 Instancias de experimentación

En este sentido, la arquitectura propuesta se fundamenta en la premisa marcada por los laboratorios remotos con arquitectura distribuida que exigen la independencia funcional de las instancias de experimentación respecto al resto de componentes, pero identificando individualmente los componentes que conforman dichas instancias y definiendo el método de conexión de los mismos. De esta forma se consigue mantener la escalabilidad del sistema sin renunciar a la replicabilidad del mismo.

Obviamente la consideración de sistemas independientes del laboratorio remoto a los componentes de las instancias de experimentación afecta al modo en el que cada uno de ellos se conecta con el servidor de laboratorio y no a la interconexión entre ellos. La Figura 3.15 muestra el esquema propuesto para la implementación de las instancias de experimentación de acuerdo a esta arquitectura.

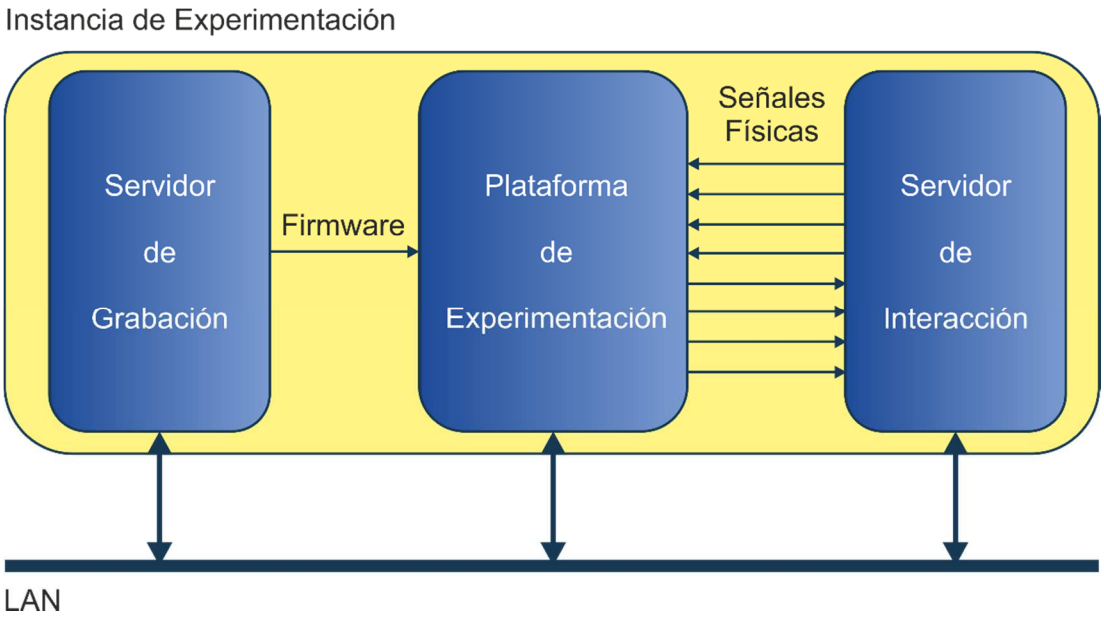

Figura 3.15. Interconexión de los componentes que forman la instancia de experimentación de acuerdo a la arquitectura abierta para el despliegue de laboratorios remotos con experimentación en sistemas embebidos.

La conexión de los componentes que forman la instancia de experimentación con el servidor del laboratorio se lleva a cabo mediante una red local, que puede ser propia del laboratorio o compartida con otros sistemas desplegados por la infraestructura. La implementación de todos estos sistemas con entidad propia permite minimizar la información transmitida por la red, que queda limitada a los comandos necesarios para el desarrollo de la experimentación. La funcionalidad de cada componente y el detalle de la información que transmite y recibe desde el servidor del laboratorio quedan reflejada en los siguientes puntos.

3.3.1.1 Servidor de grabación

El sistema de grabación del dispositivo sobre el que se desarrolla la experimentación es uno de los componentes más determinantes en la arquitectura de un laboratorio remoto desplegado para permitir experimentar sobre tecnologías embebidas. La mayoría de laboratorios remotos analizados en el capítulo anterior utilizan un programador comercial que requiere conexión directa (serie, paralelo, USB) con el servidor del laboratorio (Anzhelika, Olga, Ivanov, Sokolyanskii, & Kurson, 2015) (El-Medani, 2008) (Gomes, Patricio, Ferreira, & Costa, 2009) (Goncalves de Moraes & Mendes de Sales, 2012) (Hercog, Gergic, Uran, & Jezernik, 2007) (Hui & Tie-jun, 2008) (Rangel, Chacón, & Araque, 2011) (Sell, y otros, 2015) (Vandorpe, y otros, 2013) (Fotopoulos, Anastasios, & Anastasios, 2013) (Zenzerović & Sučić, 2011). Esto

condiciona al servidor del laboratorio a la ejecución de un software específico, complicando la replicabilidad del laboratorio y condicionando su escalabilidad.

En la arquitectura propuesta la conexión del sistema de grabación debe llevarse a cabo mediante una red de área local (Ethernet o wifi) de tal forma que la comunicación con el servidor del laboratorio se restringe al fichero a programar en el dispositivo y al resultado de su programación (Figura 3.16).

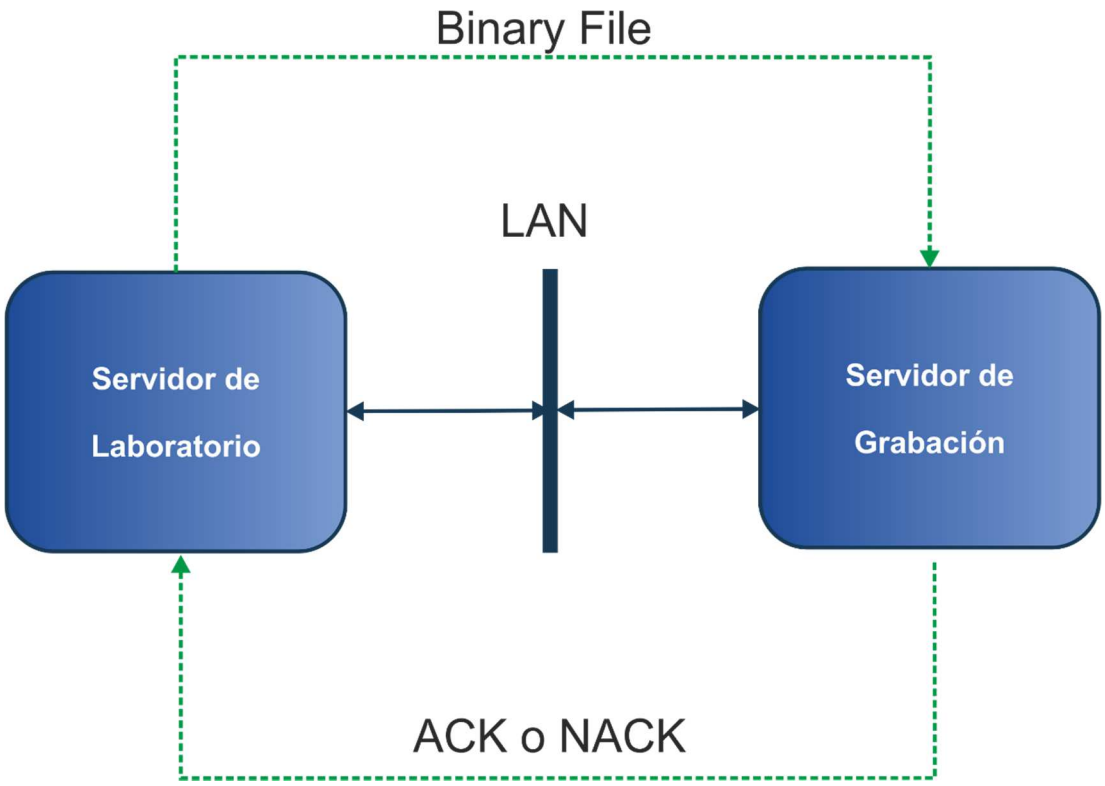

Figura 3.16. Intercambio de datos entre el servidor de grabación y el servidor del laboratorio.

Esta implementación del servidor de grabación permite remplazar la tecnología de experimentación sin realizar apenas cambios en el servidor del laboratorio y facilita enormemente la replicabilidad del laboratorio remoto.

Para muchas tecnologías de sistemas embebidos existen programadores comerciales que cumplen con los requisitos establecidos (Figura 3.17), en otras ocasiones el fabricante proporciona la información necesaria para el desarrollo a medida de un servidor de grabación mediante la misma u otra tecnología embebida (Figura 3.18) y cuando ninguna de las dos opciones es posible, el cumplimiento de la arquitectura propuesta se alcanza mediante la implementación independiente de este componente en un sistema compatible con otros programadores comerciales con conexión directa. El enorme avance llevado a cabo en los computadores de factor de

forma reducido (Krushinitskiy & Sziebig, 2013), facilita y abarata el desarrollo del servidor de programación (Figura 3.19).

Figura 3.17. Programador JTAG con conectividad ethernet para dispositivos lógicos programables de Xilinx.

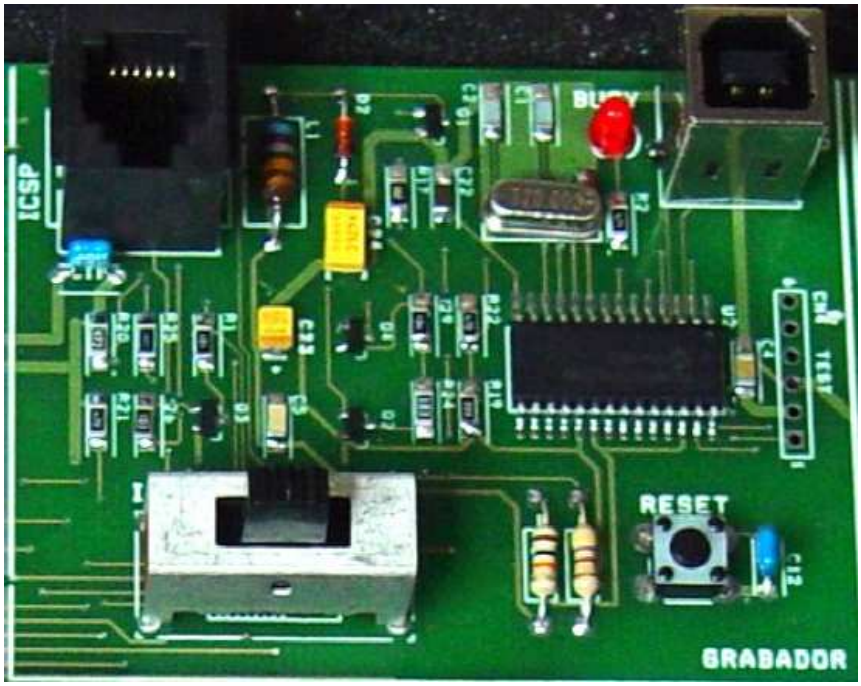

Figura 3.18. Sistema de grabación desarrollado a medida para el laboratorio remoto WebLab-PIC2 siguiendo las especificaciones de Microchip.

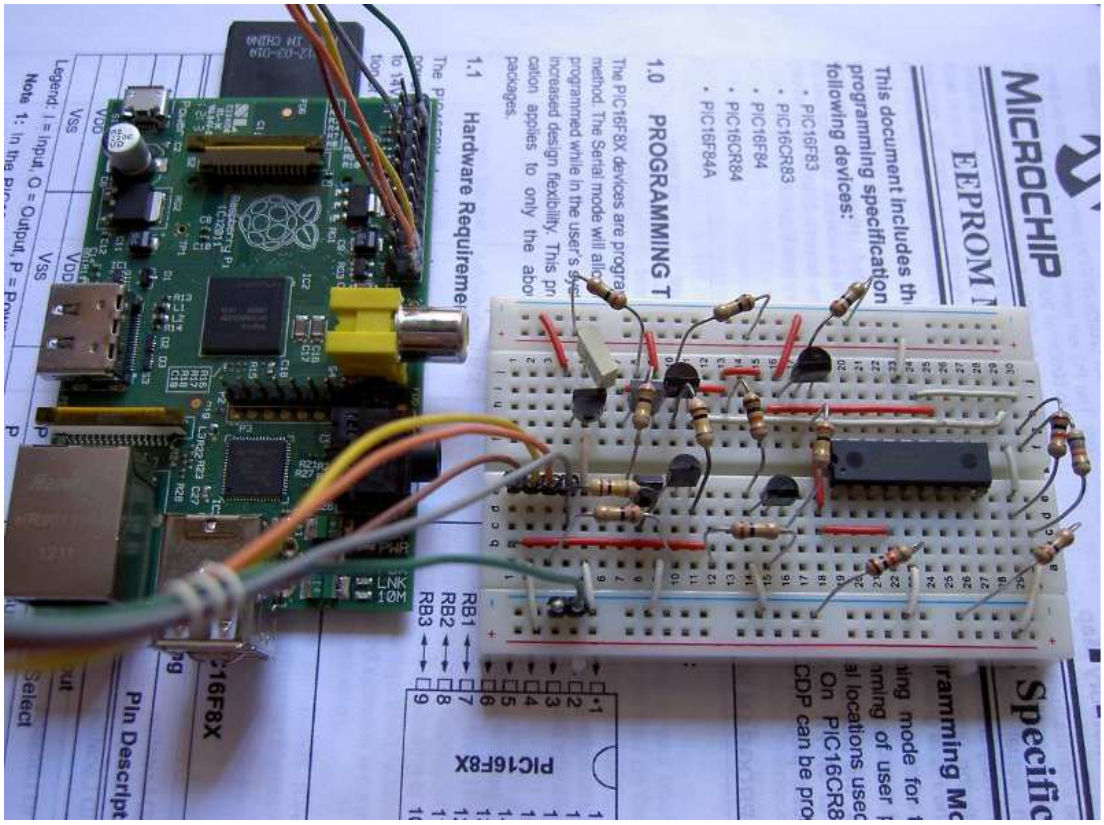

Figura 3.19. Programación de un microcontrolador PIC16F84 desde una tarjeta Raspberry Pi tipo B.

3.3.1.2 Servidor de interacción

Siguiendo la política mercada para la implementación del servidor de grabación, el servidor de interacción debe también ser desarrollado para proporcionar su funcionalidad con independencia del servidor del laboratorio.

La comunicación por medio de red local y la independencia funcional del servidor de interacción garantizan la escalabilidad y facilitan enormemente el despliegue del laboratorio al permitir disponer libremente los componentes en la instalación donde se implementa el laboratorio remoto. De esta forma los servidores de interacción de cada instancia de experimentación son accesibles desde el servidor del laboratorio a través de la red de área local, identificados unívocamente por su dirección IP. Además permite adaptar el laboratorio a nuevas tecnologías sin provocar ninguna alteración en lo referente a la interacción con el experimento, facilitando la replicabilidad.

Los computadores de factor de forma reducida (Saini, 2012) permiten la fácil implementación de estos sistemas, pero es muy importante tener en cuenta la durabilidad de la tecnología empleada para garantizar la sostenibilidad del

laboratorio y minimizar las tareas de mantenimiento. El avance en los microcontroladores de 8 bits y la integración en los mismos de transceptores Ethernet permiten también desarrollar microservidores de bajo coste que ofrecen una fiabilidad suficiente y permiten la generación de un elevado número de señales físicas de diversa naturaleza (GPIO, señales cuadradas, triangulares o sinusoidales, PWM, señales analógicas, buses de comunicación, etc.). La Figura 3.20 muestra un microservidor con conectividad Ethernet basado en un microcontrolador PIC18F67J60 de 8 bits que es capaz de implementar una pila TCP/IP completa.

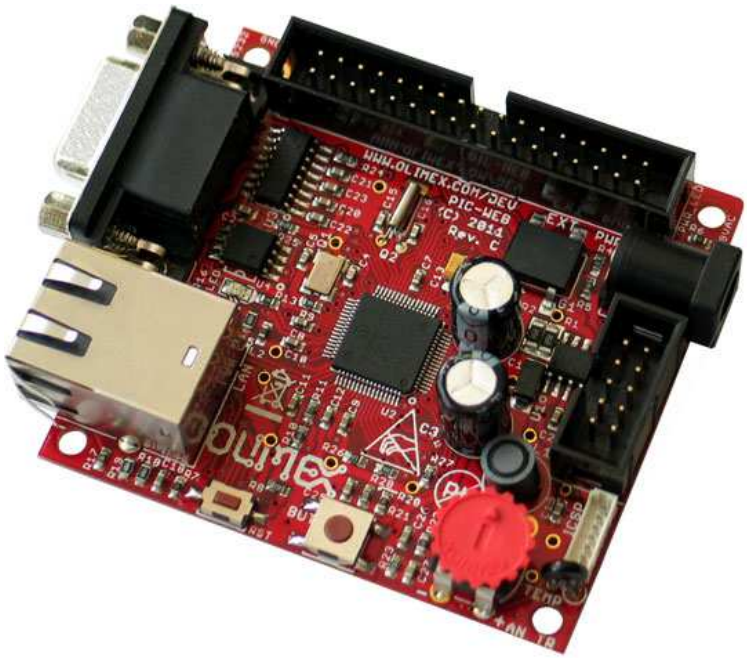

Figura 3.20. Tarjeta microservidora Olimex PIC Web, basada en el microcontrolador PIC18F97J60 de Microchip.

La implementación del servidor de interacción en sistemas embebidos con conectividad a redes de área local facilita enormemente el desarrollo del servicio que proporciona la interacción. Este servicio consiste únicamente en generar las señales físicas oportunas en los recursos hardware de las plataformas embebidas, ante la recepción de los comando HTTP enviados por el servidor del laboratorio.

Aunque, como se ha podido observar en el capítulo 2, la mayoría de laboratorios remotos optan por una cámara para proporcionar la monitorización del experimento, es necesario implementar un servicio que permita el establecimiento de una comunicación asíncrona desde el servidor de interacción hacia el servidor de laboratorio para posibilitar una monitorización virtual. En este caso, las entradas del servidor de interacción se conectarán a las salidas que deben monitorizarse de la plataforma de experimentación y se notificará al servidor del laboratorio cualquier

cambio detectado. La Figura 3.21 muestra la información compartida entre servidor de interacción y servidor de experimento.

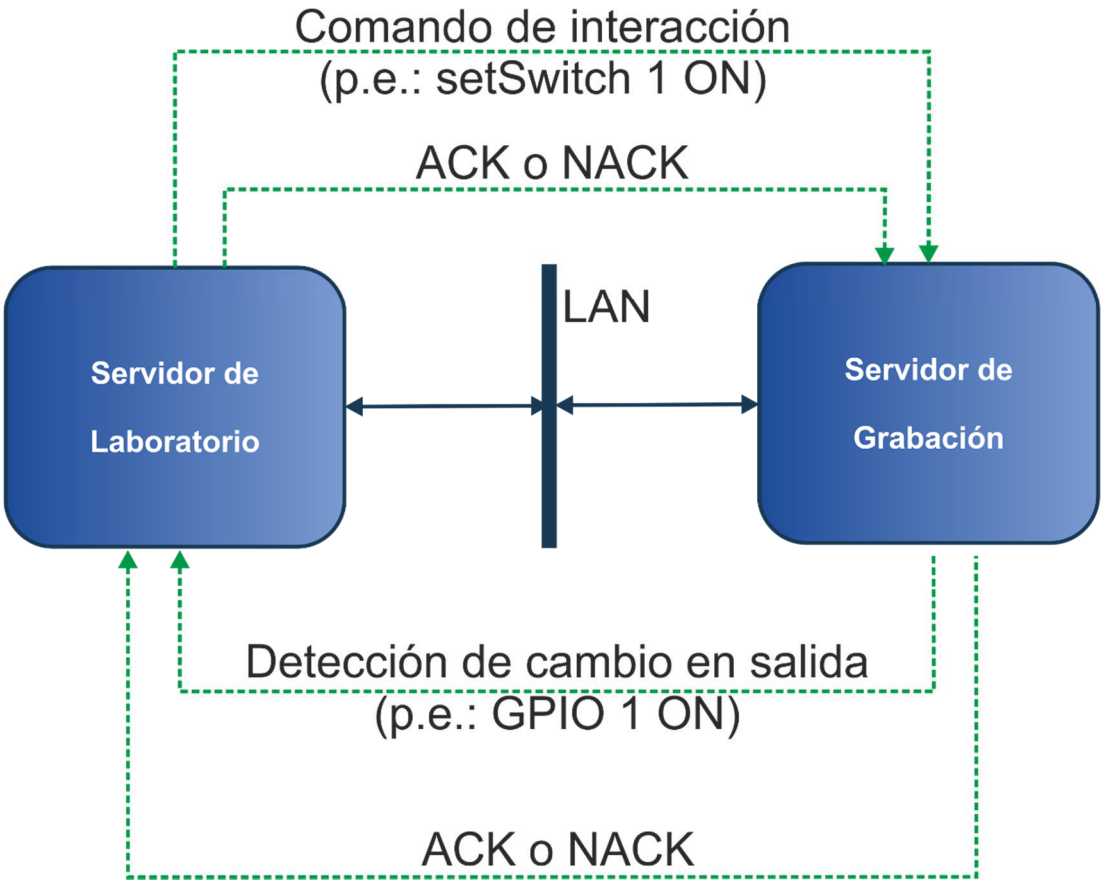

Figura 3.21. Intercambio de información para la interacción con el experimento.

3.3.1.3 Plataforma de experimentación

La plataforma de comunicación tiene conectadas las señales de entrada y/o salida con el servidor de interacción y los pines que permiten su programación con el servidor de grabación. Su conectividad a la red de área local (Figura 3.15) queda restringida a aquellos laboratorios que incluyen experimentación con este tipo de comunicaciones, permitiendo la conectividad con la red del laboratorio remoto a la que se conectan los servidores de grabación e interacción o con otra red de área local poblada por otros sistemas independientes.

Una de las principales ventajas de la arquitectura propuesta es la total independencia del laboratorio remoto con la plataforma de experimentación, permitiendo su sustitución de manera prácticamente transparente.

Una de las posibilidades más extendidas al desarrollar un laboratorio remoto para electrónica es adecuar una tarjeta de desarrollo comercial como plataforma de experimentación (Reichenbach M. , Schmidt, Pfundt, & Fey, 2011) (Lobo, 2011) (Lowe, Murray, Lindsay, & Liu, 2009) (Pop, Zutin, Auer, Henke, & Wuttke, 2011)(Vassilis Fotopoulos, 2013) (Hoang, y otros, 2015) (Hui & Tie-jun, 2008) (Butime, y otros, 2012) (Karthik, Shreya2, & Srihari, 2014) (Zenzerović & Sučić, 2011) (El-Medani, 2008) (Anzhelika, Olga, Ivanov, Sokolyanskii, & Kurson, 2015). Ese enfoque facilita la desplegabilidad del sistema en otras instituciones pero suele conllevar ciertas limitaciones, puesto que los principales recursos del dispositivo están conectados a los periféricos incluidos en la propia tarjeta de desarrollo. Esto implica que la conexión con el servidor de experimentación debe llevarse a cabo por los puertos de expansión, utilizando recursos menos habituales del dispositivo. Por este motivo, el desarrollo de una plataforma a medida para la experimentación suele proporcionar mejores resultados, al diseñar la tarjeta con los requisitos específicos del laboratorio remoto y de la experimentación que será soportada por el mismo (Morgan, y otros, 2014) (Hercog, Gergic, Uran, & Jezernik, 2007) (Henke, Ostendorff, Wuttke, & Vogel, 2012) (Gilibert, Picazo, Auer, Pester, & Cusidó, 2006). El empleo de las nuevas metodologías para la compartición de recursos hardware abiertos (Powell, 2012) facilita la replicación de circuitos electrónicos y garantiza la desplegabilidad del laboratorio en otras instituciones.

3.3.2 Servidor del laboratorio

El servidor del laboratorio es uno de los componentes más beneficiados de la arquitectura propuesta. La independencia funcional del servidor de interacciones y del servidor de grabación facilita enormemente su desarrollo, limitando su funcionalidad a la recepción de los comandos desde el cliente del laboratorio y el envío de los comandos HTTP que fuerzan su ejecución en los componentes de la instancia de experimentación.

La Figura 3.22 muestra las tareas que debe desarrollar el servidor del laboratorio, la implementación independiente de las mismas facilita la replicabilidad del laboratorio, forzando a modificar únicamente aquellas que se vean alteradas ante un cambio en la tecnología de experimentación. Por tanto los métodos fundamentales del servidor del laboratorio serán los siguientes:

- ***Grabación:*** Cuando el estudiante lanza desde el cliente una orden de grabación, el servidor del laboratorio recibe mediante un comando HTTP (POST) el firmware o fichero bitstream que contiene el experimento. El servidor del laboratorio lanza el comando de programación del dispositivo al

servidor de grabación que devuelve el resultado positivo o negativo resultante, que a su vez es enviado al estudiante.

- ***Interacción:*** Las órdenes requeridas por el estudiante desde el interfaz gráfico de usuario son recibidas mediante comandos HTTP (GET, PUT o POST) y transformadas en comandos que son directamente enviados hacia el servidor de interacción.
- ***Monitorización:*** En aquellos laboratorios remotos cuyo interfaz gráfico de usuario dispone de componentes virtuales para la monitorización del experimento es fundamental que el servidor del laboratorio disponga de un espacio de memoria consumible desde el cliente en el que se actualice el valor de las señales monitorizadas ante cada alteración enviada por el servidor de interacción.
- ***Registro de Sesión:*** Todas las acciones llevadas a cabo por el estudiante deben ser registradas por el servidor de autenticación para posteriormente proporcionar al profesor la información necesaria para la evaluación de las sesiones de experimentación.

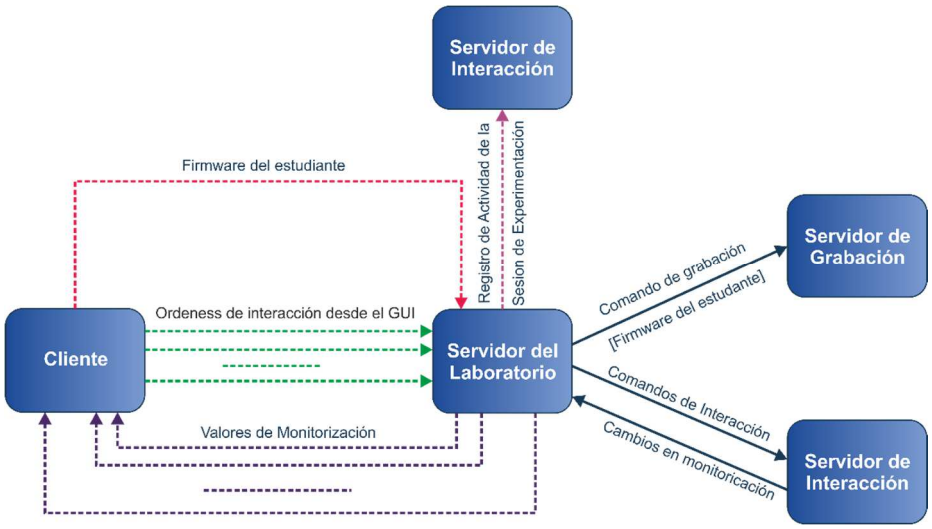

Figura 3.22. Actividad del servidor del laboratorio en la arquitectura abierta para el despliegue de laboratorios remotos en sistemas embebidos.

3.3.3 Cliente

El desarrollo del sistema que proporciona el interfaz de usuario al estudiante también se ve simplificado con la adopción de esta arquitectura. La similitud y simplicidad de la experimentación con las diferentes tecnologías y dispositivos de sistemas embebidos permiten el desarrollo de un cliente basado únicamente en tecnologías

web de forma sencilla y eficiente. Esto garantiza la universalidad del laboratorio remoto (Orduña, García-Zubia, Irurzun, & Rodriguez-Gil, 2011).

La Figura 3.22 muestra el flujo de información que se establece entre el cliente y el servidor del laboratorio. El desarrollo de las sesiones de experimentación puede ser ejecutado desde un único interfaz gráfico de usuario en el que se monitoriza el estado de la plataforma de experimentación mediante una cámara o un componente virtual y se generan las órdenes de interacción desde dispositivos virtualizados. El aspecto y disposición de los componentes virtuales debe representar en el mayor grado posible el mismo dispuesto en un sistema de desarrollo para experimentación presencial para optimizar la inmersión del estudiante en el laboratorio (Corter, Nickerson, Esche, & Chassapis, 2004) (Nickerson, Corter, Eschea, & Chassapis, 2007).

3.3.4 Servidor de administración

El servidor de las tareas de administración no se ve afectado por el empleo de la arquitectura propuesta. Si bien, aunque no es necesario, las características de desplegabilidad, escalabilidad, replicabilidad y universalidad proporcionadas por esta arquitectura requieren el desarrollo de muchas funcionalidades (soporte para múltiples métodos de autenticación, balanceo de carga entre instancias de experimentación, tracking de sesión, etc.) en el servidor de autenticación, siendo recomendable la integración con sistemas de gestión de laboratorios remotos, que proporcionan componentes software a medida para soportar esas características y garantizan la sostenibilidad del laboratorio remoto. Además estos sistemas RLMS facilitan la compartición del laboratorio con otras instituciones (Orduna, Rodriguez-Gil, & Lopez-de-Ipina, 2012).

3.3.5 Esquema general de la arquitectura propuesta

Las arquitecturas de todos los laboratorios remotos par la experimentación con sistemas embebidos analizados en el capítulo 2 presentan dependencias que comprometen la desplegabilidad, la escalabilidad o la replicabilidad.

Muchos laboratorios remotos se han diseñado intentando replicar la arquitectura de los sistemas de desarrollo que permiten la experimentación deseada en los laboratorios presenciales (Lobo, 2011) (Pop, Zutin, Auer, Henke, & Wuttke, 2011) (Vassilis Fotopoulos, 2013) (Hui & Tie-jun, 2008) (Vandorpe, y otros, 2013) (Butime, y otros, 2012) (Karthik, Shreya2, & Srihari, 2014) (Goncalves de Moraes & Mendes de

Sales, 2012) (Zenzerović & Sučić, 2011) (Rangel, Chacón, & Araque, 2011) (El-Medani, 2008) (Ferreira, Nedić, Machotka, Nafalski, & Göl, 2010) (Tawfik, Sancristobal, Martin, Diaz, & Castro, 2012) (Anzhelika, Olga, Ivanov, Sokolyanskii, & Kurson, 2015). Aunque estos laboratorios son fácilmente desplegables debido a la utilización de sistemas comerciales para su desarrollo, estos laboratorios presentan una importante dependencia con los dispositivos embebidos sobre los que se desarrolla la experimentación y dificultan la publicación de nuevas instancias comprometiendo la replicabilidad y la escalabilidad.

Algunos laboratorios remotos se centran en el desarrollo de un determinado conjunto de experimentos (Morgan, y otros, 2014) (Limpraptono, Sudibyo, Ratna, & Arifin, 2011) (Kutlu & Taşdelen, 2010), requiriendo la experimentación con una determinada tecnología embebida sin capacidad de replicabilidad y exigen el desarrollo de componentes hardware a medida del experimento que dificultan la desplegabilidad.

Otros laboratorios (Reichenbach M. , Schmidt, Pfundt, & Fey, 2011) (Lowe, Murray, Lindsay, & Liu, 2009) (Hercog, Gergic, Uran, & Jezernik, 2007) (Hoang, y otros, 2015) (Gilibert, Picazo, Auer, Pester, & Cusidó, 2006) (Sell, y otros, 2015) permiten escalar el número de instancias comprometiendo la replicabilidad a nuevas tecnologías y dificultando el despliegue de los laboratorios remotos en otras instituciones.

Por último hay laboratorios remotos (Henke, Ostendorff, Wuttke, & Vogel, 2012) centrados en las características de los dispositivos a controlar mediante las tecnologías embebidas. Estos laboratorios limitan la naturaleza de la experimentación restringiendo los experimentos soportados.

En definitiva los laboratorios remotos para experimentación en sistemas embebidos analizados se han desarrollado en base a la experimentación sobre un determinado sistema de desarrollo o tecnología embebida, o a la provisión de un determinado conjunto de experimentos mediante diferentes tecnologías embebidas. Ningún laboratorio ha basado su arquitectura en la naturaleza de los sistemas embebidos atendiendo a las afinidades de las distintas tecnologías existentes para su desarrollo.

La principal hipótesis de este trabajo es demostrar cómo la identificación de los principales componentes necesarios para el desarrollo de experimentación sobre tecnologías embebidas, su desarrollo independiente con funcionalidad propia, y finalmente su interconexión a través de la red local del laboratorio remoto proporcionan las principales características que se han definido para la

implementación de un laboratorio remoto para experimentación sobre sistemas embebidos:

- Desplegabilidad.
- Escalabilidad.
- Replicabilidad.

La Figura 3.23 muestra la disposición de todos los componentes que componen la arquitectura abierta de fácil despliegue para permitir la experimentación con sistemas embebidos desde laboratorios remotos.

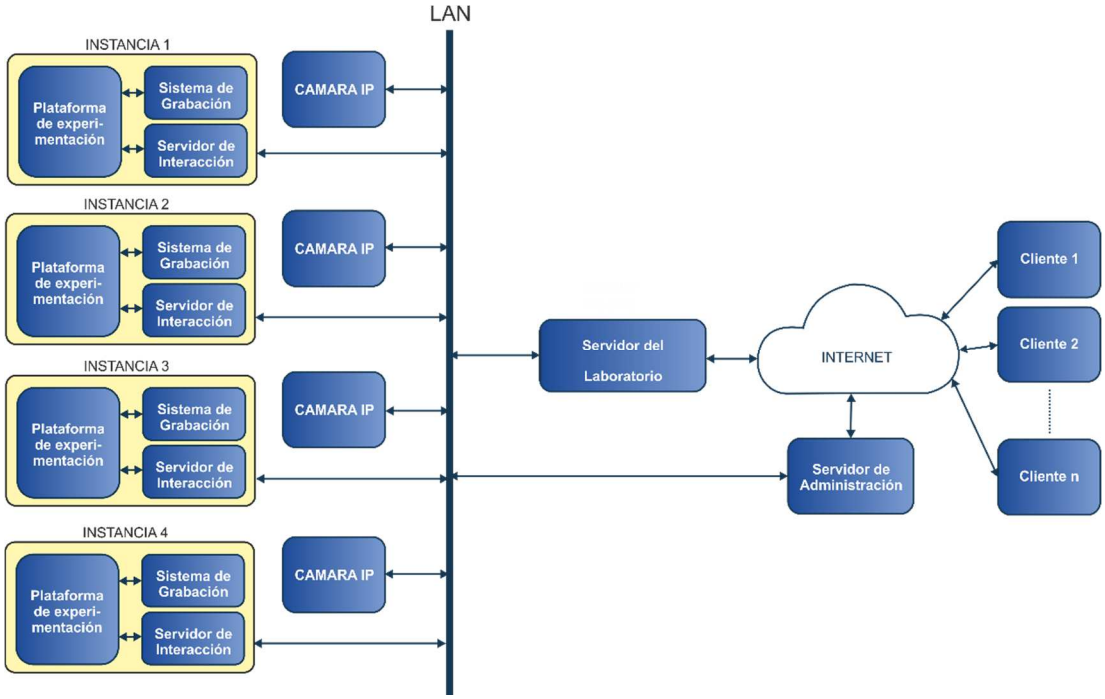

Figura 3.23. Esquema general de la arquitectura abierta para el despliegue de laboratorios remotos en sistemas embebidos.

En el siguiente capítulo se analizarán diferentes implementaciones de laboratorios remotos basados en esta arquitectura para la experimentación con diferentes tecnologías de sistemas embebidos, para posteriormente en el capítulo 5 evaluar estas implementaciones y validar el cumplimiento de las características definidas en cada uno de los casos.

Capítulo

4

Implementación de la arquitectura propuesta

A lo largo de este capítulo y siguiendo las especificaciones establecidas durante la descripción de la arquitectura se describirán cuatro implementaciones diferentes de laboratorios remotos para experimentación con diferentes tecnologías para el desarrollo de sistemas embebidos. Aunque las instalaciones de WebLab-Deusto disponen de un número superior de laboratorios remotos en tecnologías embebidas, el propósito de este capítulo no es la mera descripción de los mismos sino demostrar la adecuación de la arquitectura propuesta en el capítulo 3 a distintas tecnologías y especificaciones, y constatar su validez en las diferentes implementaciones. Por eso se han seleccionado cuatro laboratorios remotos diferentes que cubren desde un sistema fundamentado en el bajo coste del laboratorio remoto, hasta un sistema que permite la experimentación remota de estudiantes con maquetas industriales de alto valor económico.

Las cuatro implementaciones que se describirán en este capítulo son las siguientes:

1. ***WebLab-PIC***: Laboratorio remoto de bajo coste y fácil despliegue para la experimentación con microcontroladores PIC

2. ***WebLab-Robot***: Laboratorio remoto para la experimentación con un robot móvil.

3. ***WebLab-Box***: Plataforma profesional para facilitar el despliegue de laboratorios remotos para experimentación con sistemas embebidos.

4. ***WebLab-Elevator***: Laboratorio remoto para la experimentación con lógica programable sobre una maqueta industrial que emula un ascensor de tres pisos.

Para cada una de estos sistemas se analizará la implementación de cada uno de los componentes de la arquitectura propuesta indicando las herramientas hardware/software utilizadas durante su desarrollo.

Además, se analizará el desarrollo de la experimentación ofrecida por cada laboratorio remoto, describiendo las tareas que debe realizar el estudiante durante una sesión.

El cometido de este capítulo es aportar los logros técnicos que han permitido implementar un conjunto de laboratorios remotos siguiendo la arquitectura descrita en el capítulo 3. La validación de los objetivos estipulados en cada uno de ellos se pospone al capítulo 5, en el que se evaluará el cumplimiento de las características establecidas para cada implementación.

La estructura que se seguirá para la descripción de cada implementación será la siguiente:

1. Desarrollo e interconexión de cada uno de los componentes incluidos en la arquitectura propuesta.
    - Servidor de grabación.
    - Servidor de interacción.
    - Plataforma de experimentación.
    - Servidor del laboratorio.
    - Cliente.
    - Servidor de administración.

2. Descripción procedimental de una sesión de experimentación.

3. Resumen del laboratorio remoto concretando los aportes del mismo respecto a la evaluación de la arquitectura planteada.

4.1 Laboratorio remoto WebLab-PIC

El laboratorio remoto WebLab-PIC ha sido implementado como una prueba de concepto para demostrar el correcto funcionamiento de la arquitectura propuesta en un laboratorio remoto de mínimo coste que resultara fácilmente desplegable en cualquier instalación.

Es por tanto un laboratorio remoto peculiar que cuenta con la singularidad de ser el laboratorio remoto más pequeño y económico existente para el desarrollo de experimentación con tecnologías embebidas (García-Zubia, Angulo, Hernández-Jayo, & Orduña, 2008) (García-Zubia, Angulo, Hernández, & Orduña, 2008) (García-Zubia, y otros, 2010). Pese a ello, dispone de todos los componentes individuales identificados en la arquitectura y permite el desarrollo de experimentación completa sobre un microcontrolador de 8 bits (Figura 4.1).

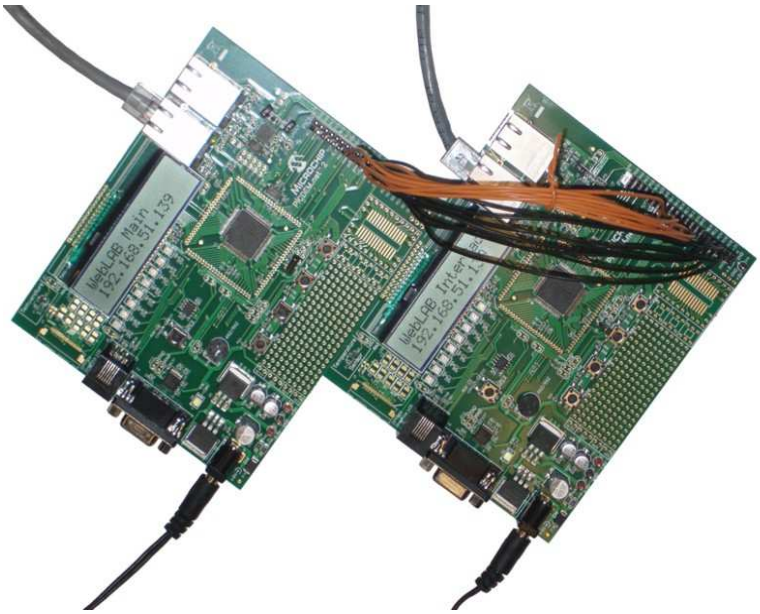

Figura 4.1. Laboratorio remoto WebLab-PIC desplegado en Budapest durante el congreso "7th Workshop on Microelectronics Education in Europe (EWME 2008)".

La implementación de este dispositivo se hizo posible gracias al avance en el rendimiento de los microcontroladores de 8 bits así como a su integración de nuevas tecnologías de comunicaciones. La aparición en el mercado de un microcontrolador de 8 bits de alto rendimiento con un transceptor de ethernet integrado, la familia de microcontroladores PIC18FX7J60 de Microchip (Microchip Technology, 2011) (Figura 4.2), permitió concebir el desarrollo de todos los componentes identificados en la arquitectura como necesarios para el despliegue de un laboratorio remoto en sistemas embebidos en el propio firmware del mismo dispositivo sobre el que se desarrollará la experimentación.

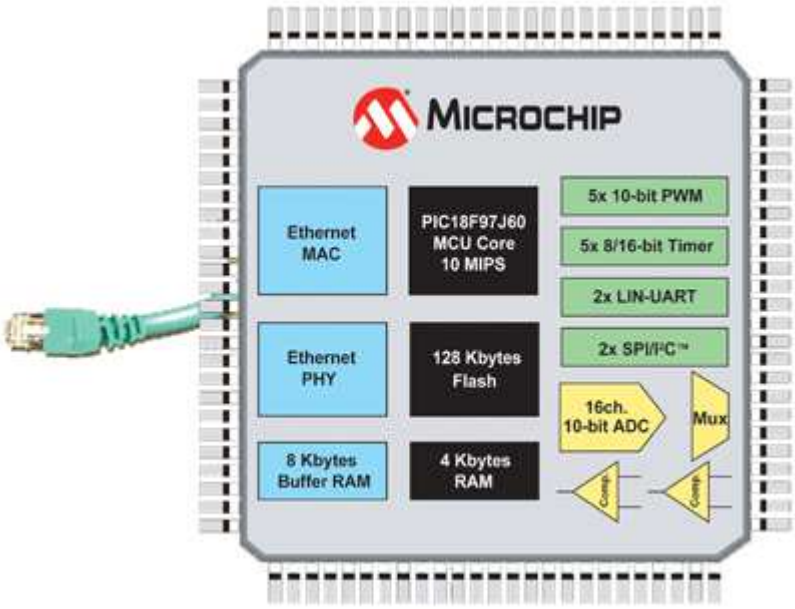

Figura 4.2. Características del microcontrolador PIC18F97J60.

Los limitados recursos del microcontrolador sobre el que se desarrolla la experimentación (Figura 4.2), impiden la implementación de todos los componentes necesarios para el despliegue del laboratorio remoto sobre un mismo dispositivo. De este modo, los componentes que conforman este laboratorio se distribuyen sobre dos microcontroladores PIC18F97J60 idénticos (Figura 4.3).

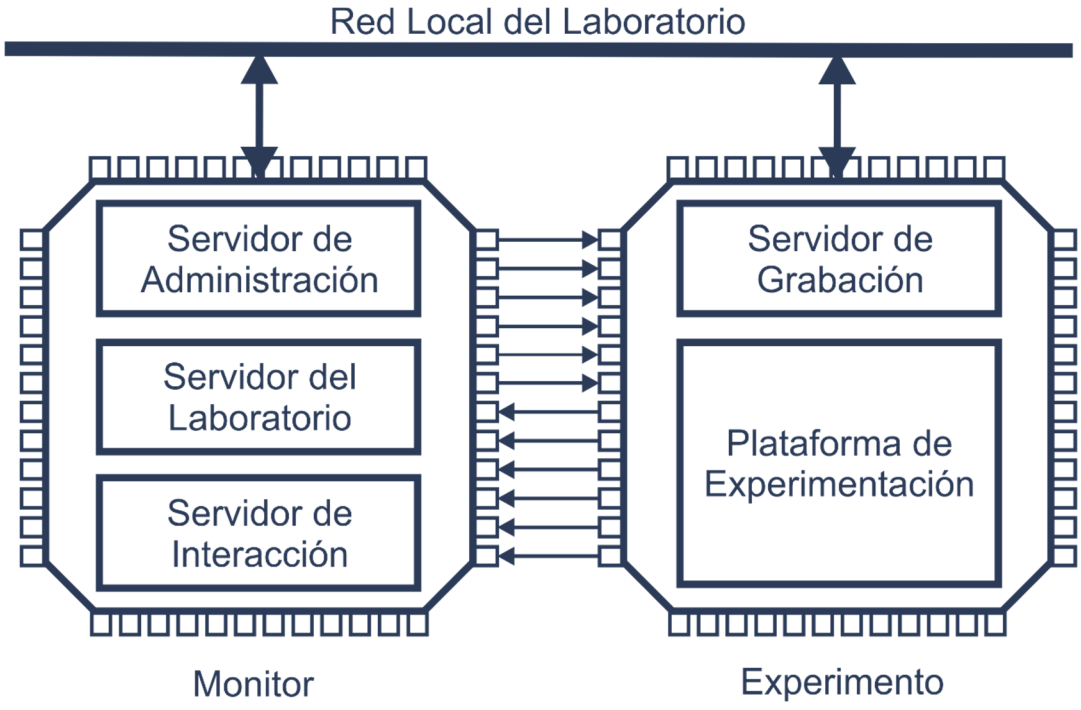

Figura 4.3. Distribución de los componentes del laboratorio remoto WebLab-PIC.

El microcontrolador "monitor" se encarga del desarrollo de las sesiones y contiene los componentes servidor de administración, servidor del experimento y servidor de interacción. Por otro lado el microcontrolador "experimento" alberga al bootloader que implementa el servidor de grabación y representa la propia plataforma de experimentación en la que se programará el firmware desarrollado por el estudiante.

4.1.1 Implementación del laboratorio remoto WebLab-PIC

Dado que todos los componentes de la arquitectura van a ser implementados en el firmware del microcontrolador sobre el que se va a desarrollar la experimentación, únicamente fue necesaria una herramienta software de desarrollo para la implementación de la totalidad del laboratorio remoto. El sistema de desarrollo que Microchip proporciona para la implementación de sistemas embebidos con los microcontroladores que produce es el entorno de desarrollo integrado MPLAB. Esta herramienta permite la integración de compiladores C, como el MPLAB-C 18 que fue utilizado para la programación de este laboratorio remoto.

Con el lanzamiento de la familia PIC18FJX7J60 Microchip publicó la implementación de una pila TCP/IP completa, compatible con el compilador seleccionado para el desarrollo de este sistema (Microchip Technology, 2008). En la Figura 4.4 se pueden observar los protocolos implementados en la pila TCP/IP desarrollada por Microchip. Los protocolos TCP y UDP están implementados en la capa de transporte y los protocolos DHCP, HTTP y TFTP en la capa de aplicación. Estos protocolos serán fundamentales en el desarrollo de este laboratorio remoto.

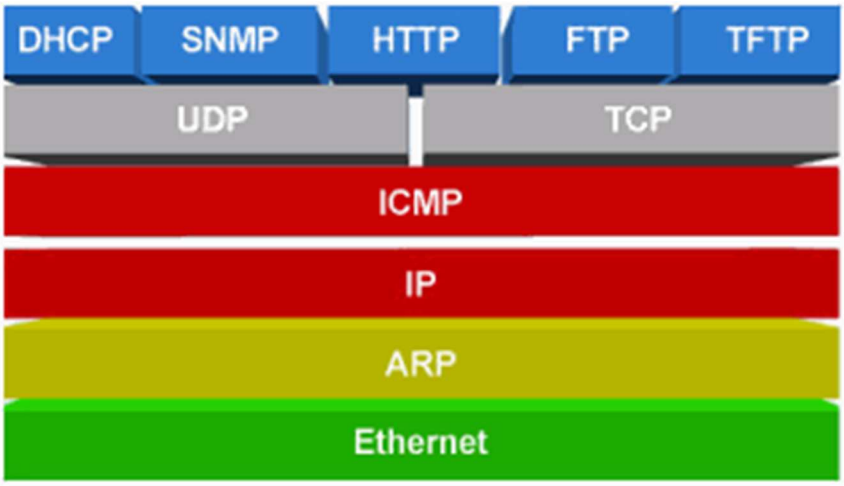

Figura 4.4. Resumen de las diferentes capas de la pila TCP/IP publicada por microchip para su integración en microcontroladores PIC18FJX7J60.

A continuación se describe la implementación de los diferentes componentes del laboratorio remoto de acuerdo a la arquitectura abierta de fácil despliegue para permitir la experimentación con sistemas embebidos desde laboratorios remotos.

4.1.1.1 Servidor de grabación

Los microcontroladores de la familia PIC18F de microchip disponen de un sistema propio de programación (Microchip Technology, 2011). Estos dispositivos disponen de dos métodos de funcionamiento, "ejecución" en el cual simplemente ejecutan las instrucciones almacenadas en su memoria de programa, y "monitor" que permite la escritura física de la memoria de programa a través de un protocolo serie síncrono definido a medida. Aunque la tensión de trabajo de estos microcontroladores oscila entre 2 a 6 Voltios, la forma de seleccionar el modo monitor para llevar a cabo la programación es introducir una tensión de 13,2 Voltios a través de su pin MCLR'. Una vez seleccionado este modo, Microchip proporciona una serie de sencillos comandos que permiten la grabación del dispositivo mediante un protocolo serie propietario en el que los datos se introducen mediante el pin PWD y la sincronización a través del pin PWC.

La utilización de un dispositivo grabador externo quedo inmediatamente descartada al ir en contra de la filosofía establecida para el desarrollo de este laboratorio en el que todos los componentes debían estar integrados en el mismo dispositivo. La única alternativa que cumplía con tal requisito para la implementación del servidor de grabación quedó limitada a la utilización de un bootloader. En el momento en el que este laboratorio fue desarrollado, los únicos bootloaders disponibles para estos microcontroladores recibían el código máquina por comunicación serie, incumpliendo con la exigencia de la arquitectura planteada de disponer carácter independiente y conectividad mediante una red de área local. Por tanto fue necesario el desarrollo de un bootloader a medida para la implementación de este componente.

La primera aproximación del sistema de grabación fue implementada utilizando el protocolo File Transfer Protocolo (FTP) para la transferencia del fichero que contiene el firmware con el que programar el microcontrolador. Pese al correcto funcionamiento del bootloader desarrollado, tuvo que ser descartado por consumir demasiados recursos del microcontrolador. La implementación del protocolo TCP, necesario para soportar la transferencia de ficheros por FTP y las limitaciones del propio microcontrolador (Figura 4.4) conllevan un consumo de las memorias de programa y datos del dispositivo cercano al 50% del total (Figura 4.5). Además debido a las comprobaciones de integración del fichero incluidas por el protocolo FTP, los

tiempos de transferencia y programación resultaban demasiado altos para su utilización en un laboratorio remoto.

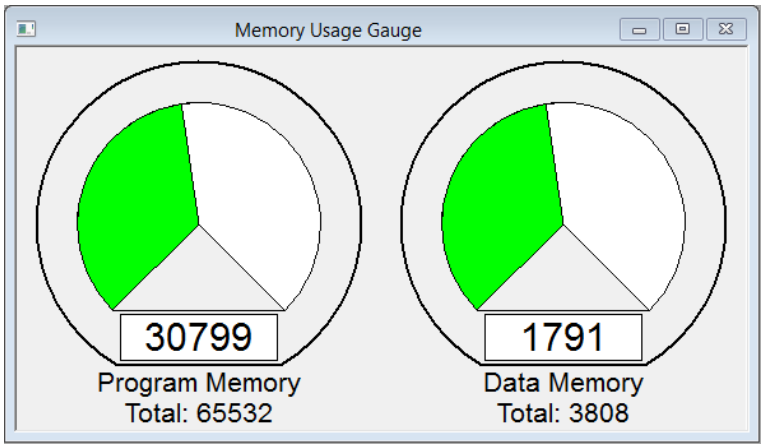

Figura 4.5. Recursos de memoria consumidos por un bootloader para el microcontrolador PIC18F97J60 implementado mediante protocolo FTP.

Como alternativa a este protocolo se desarrolló un segundo bootloader utilizando en esta ocasión el protocolo Trivial File Transfer Protocol (TFTP). Este protocolo permite la simple de transferencia de archivos entre un cliente y un servidor. TFTP carece de la seguridad y de la mayoría de las características avanzadas que ofrecen protocolos de transferencia de archivos más robustos, tales como FTP. Sin embargo su sencillez, al utilizar el protocolo User Datagram Protocol (UDP) en lugar de TCP, permite la implementación de un bootloader con unos tiempos de programación y un consumo de recursos de memoria de programa mucho más ajustados. La Figura 4.6 muestra como el bootloader no llega a consumir 4KWords, de los 64KWords con los que cuenta el microcontrolador. Hay que tener en cuenta que la mayoría de las instrucciones máquina de los microcontroladores de la familia PIC18 de Microchip ocupan 16 bits, con los que los 128KBytes de memoria FLASH de programa se traducen en 64KWords disponibles.

Toda la programación de este bootloader está orientada hacia el consumo de la memoria de datos en lugar de la memoria de programa. Las múltiples constantes necesarias para la implementación de la pila, así como las direcciones lógicas y físicas son almacenadas en memoria de datos para minimizar la memoria de programa consumida. Esto es así porque la memoria de datos de un bootloader no afecta al programa que posteriormente se encargará de recibir, almacenar en memoria de programa y lanzar su ejecución. Cuando el bootloader finaliza la carga del programa y lanza su ejecución, el programa recibido utilizará su propio mapa de memoria de datos reescribiendo toda la información mantenida por el bootloader reiniciando completamente el contexto de aplicación.

Por otro lado es fundamental que el bootloader desarrollado no utilice interrupciones, reservando este recurso para su gestión por el estudiante.

Aunque en el momento del desarrollo de este laboratorio remoto Microchip no proporcionaba ninguna nota de aplicación que contuviera un bootloader basado en protocolo TFTP para su familia de microcontroladores PIC18FX7J60, la oficina técnica de Microchip en España nos proporcionó todo el soporte necesario para el desarrollo de este componente.

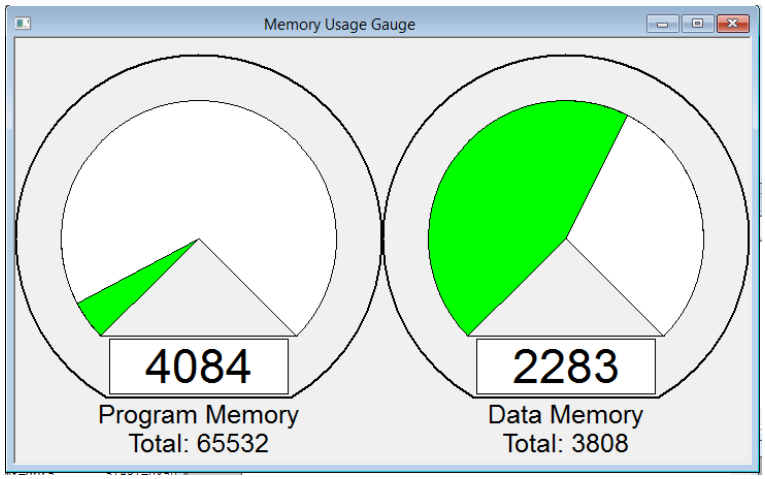

Figura 4.6. Recursos de memoria consumidos por un bootloader para el microcontrolador PIC18F97J60 implementado mediante protocolo TFTP.

Aunque generalmente un programa bootloader se desarrolla para provocar una única programación del dispositivo, en el caso de un laboratorio remoto la programación del dispositivo será requerida en cada sesión de un estudiante. Para eso es necesario localizar un método que permita pasar del modo ejecución al modo bootloader recursivamente.

Como se ha mencionado en el apartado 3.1.4, el desarrollo de un bootloader como servidor de grabación de un laboratorio remoto debe ir dirigido a permitir la experimentación de los estudiantes sin conllevar modificaciones en el código que estos generan respecto al desarrollado para un laboratorio presencial. Evitar configuraciones específicas motivadas por el uso del laboratorio es una tarea complicada. Una alternativa posible es desarrollar un fichero linker script a medida, con la redefinición de los espacios de memoria de acuerdo al bootloader, que el alumno deberá importar en su proyecto cada vez que desarrolla un proyecto para el laboratorio remoto. Esta solución presenta dos problemas:

1. *Limita la herramienta de compilación.* Existen multitud de herramientas de compilación para microcontroladores PIC y no todas ellas permiten

configurar el fichero linker script. Además este enfoque impide el desarrollo en ensamblador algo aún hoy en día común en microcontroladores de 8 bits.

2. *Supone en sí misma una modificación al proceso de desarrollo convencional.* Incluir pasos en el desarrollo del sistema exclusivos en la experimentación sobre el laboratorio remoto puede provocar confusión en el estudiante si no asimila correctamente el fundamento de las modificaciones impuestas.

Para solucionar estos problemas se ha desarrollado un bootloader basado en el análisis y reposicionamiento directo de las instrucciones ubicadas entre vectores iniciales. Este bootloader supone en sí mismo uno de los logros de investigación aportados por esta tesis.

4.1.1.1.1 Gestor de arranque flash para la reubicación dinámica del código genérico de un microcontrolador

La alternativa transparente al estudiante es alterar automáticamente el posicionamiento original de las instrucciones máquina enviado por el estudiante para conseguir la adecuación a los vectores impuestos por el bootloader. De acuerdo a la arquitectura planteada, este meta-reposicionamiento del código puede realizarse en el servidor del laboratorio o en el propio bootloader. En laboratorios remotos en los que el servidor del laboratorio se implementa sobre un sistema de alto rendimiento su integración en este componente suele resultar más sencillo debido al empleo de herramientas de desarrollo de alto nivel. En el caso de este laboratorio remoto, dado que todos los elementos del mismo se van a implementar sobre el mismo dispositivo sobre el que se desarrolla la experimentación, la opción más lógica es integrar el código de reposicionamiento en el propio bootloader, eliminando las dependencias tecnológicas del servidor del laboratorio y mejorando la replicabilidad de este componente.

Para ello es importante conocer el mapa de memoria de programa del dispositivo de experimentación. La arquitectura de los microcontroladores de la familia PIC18F cuenta con una memoria de programa lineal y dispone de un bus de direcciones de 21 bits. Esto permite una dimensión máxima de la memoria de 2MBytes, que en el caso del microcontrolador PIC18F97J60 queda restringido a 64 KBytes. Sobre este mapa de memoria el los microcontroladores de la famila PIC18F, albergan tres vectores de memoria que resultan fundamentales en el desarrollo de la lógica de reposicionamiento.

1. Vector de reset: Posición de la memoria de programa que alberga la primera instrucción que se ejecuta tras la provocación de un reset. Es la posición 0.

2. Vector de interrupciones de alta prioridad: Posición que se carga en el contador del programa (PC) cuando se detecta una fuente de interrupción activa y configurada como de alta prioridad. Es la posición 8h.

3. Vector de interrupciones de baja prioridad: Posición que se carga en el contador del programa (PC) cuando se detecta una fuente de interrupción activa y configurada como de baja prioridad. Es la posición 18h.

La linealidad de la memoria de programa de los microcontroladores PIC facilita enormemente el trabajo de los compiladores. Cumpliendo con el libro blanco del fabricante (Microchip Technology, 2010), los principales compiladores desarrollados para microcontroladores PIC (HI-TECH, 2003) (Copyright Custom Computer Services, 2015) limitan el código incluido en los vectores a saltos con posicionamiento directo hacia las rutinas de gestión correspondientes. Sin embargo la posición de los vectores de interrupciones después del vector de reset impide el desplazamiento relativo del código del estudiante, pudiendo este utilizar o no interrupciones. El problema por tanto queda limitado a las instrucciones que van desde el vector de reset (pos 0h) hasta el vector de interrupciones de alta prioridad (8h) (Figura 4.7). La solución adoptada es colocar en el vector de reset un salto con direccionamiento directo a la posición donde comienza el bootloader (goto 1DC00h) y recolocar las instrucciones alojadas entre los vectores iniciales a las posiciones libres dejadas detrás al finalizar el bootloader (1FB08). Es necesario convertir todos los saltos relativos incluidos en esas 8 primeras posiciones a saltos absolutos.

De esta forma, tanto al alimentar el microcontrolador (Power on Reset) como al provocar un reset por código o a través de la línea habilitada para ello (MCLR'), el microcontrolador pasa a modo bootloader. Lógicamente el paso de modo bootloader a modo ejecución se lleva a cabo cuando el bootloader finaliza la programación de la memoria de programa con el programa del estudiante.

La única limitación impuesta por este bootloader es que el estudiante pasa a disponer únicamente de 60KBytes para su programa, en lugar de los 64 originales.

Además de mejorar la inmersión del laboratorio, ya que permite al estudiante desarrollar el experimento exactamente igual que como lo lleva a cabo en el laboratorio presencial, el diseño de este servidor de grabación es fácilmente adaptable a otros fabricantes y familias de microcontroladores proporcionando un gran factor de replicabilidad al laboratorio remoto.

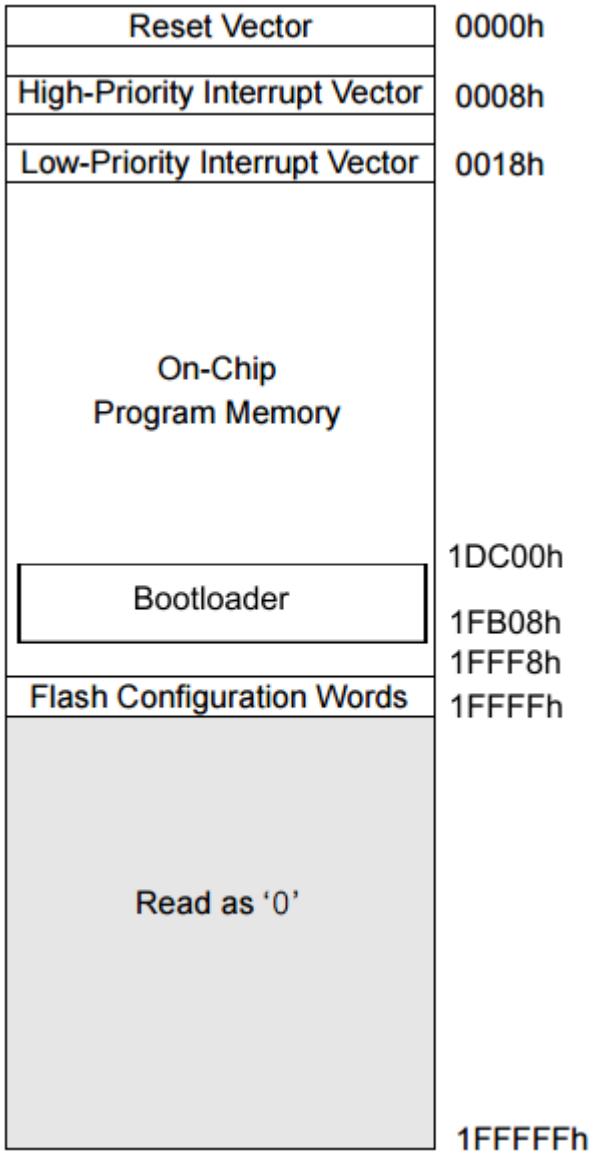

Figura 4.7. Mapa de memoria del PIC18F97J60 con el bootloader desarrollado como sistema de grabación integrado.

El funcionamiento del bootloader se observa en el diagrama de flujo de la Figura 4.8.

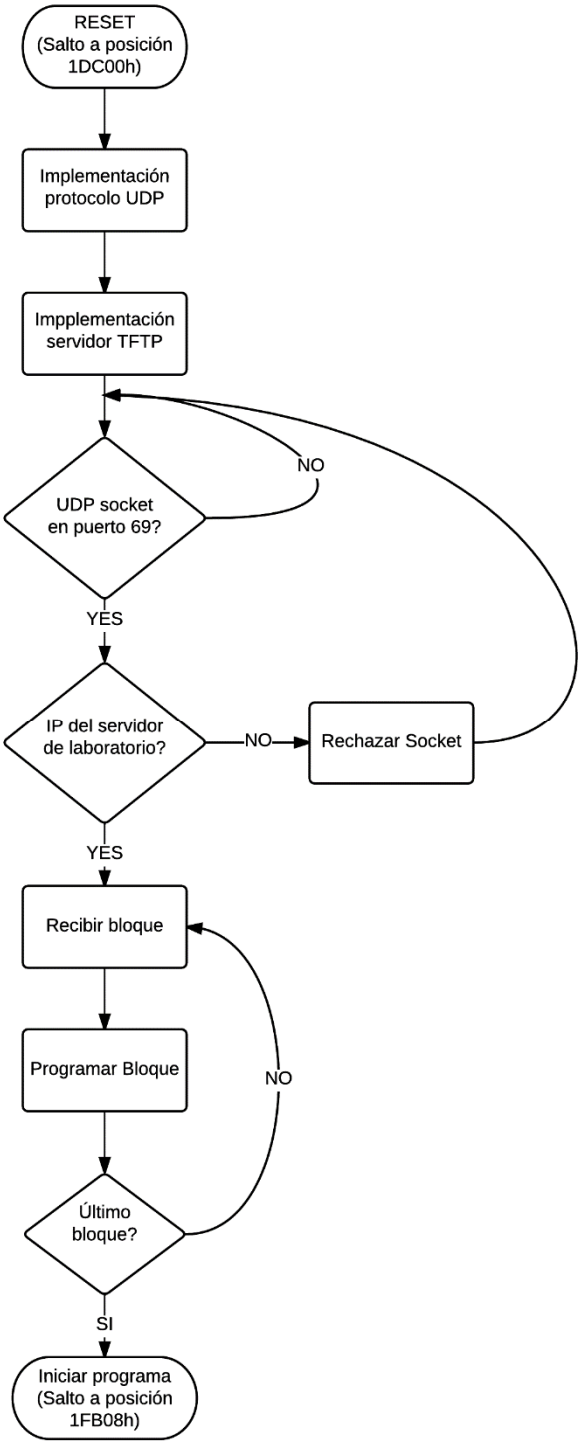

Figura 4.8. Diagrama de flujo correspondiente al bootloader del laboratorio remoto WebLab-PIC.

La función que lleva a cabo la programación de cada bloque de instrucciones incluido en el firmware enviado por el estudiante se puede observar en la Tabla 4.1.

Tabla 4.1. Función encargada de escribir los bloques de códigos contenidos en el firmware de experimentación en el bootloader del laboratorio remoto WebLab-PIC.

```
static BOOL RawWriteFlashBlock(DWORD Address, BYTE *BlockData, BYTE
*StatusData){
	BYTE i;
	WORD w;
	BOOL WriteNeeded = FALSE;
	BOOL WriteAllowed = TRUE;
	BOOL VerifyPassed = TRUE;

	//Carga el bloque de instrucciones en la estructura de datos TABLAT
	TBLPTR = Address;
	for(w = 0; w < FLASH_WRITE_SIZE; w++)  {
		_asm TBLRD _endasm
		if(TABLAT != 0xFF)
			WriteAllowed = FALSE;
		i = BlockData[w];
		if(TABLAT != i){
			TABLAT = i;
			WriteNeeded = TRUE;
		}
		_asm TBLWTPOSTINC _endasm
	}
	TBLPTR = Address;
	// Lleva a cabo la grabación sin verificación
	if(StatusData != NULL)
		memset(StatusData, 0xFF, FLASH_WRITE_SIZE>>3);

	if(WriteNeeded){
		if(WriteAllowed){
			ClrWdt();
			// Grabación de la memoria FLASH
			EECON1bits.FREE = 0;
			EECON1bits.WREN = 1;
			_asm
				movlw  0x55
				movwf  EECON2, ACCESS
				movlw  0xAA
				movwf  EECON2, ACCESS
				bsf          EECON1, 1, ACCESS   //WR
			_endasm
			EECON1bits.WREN = 0;
			ClrWdt();
		}
	}
}
```

En la Tabla 4.2 se muestra el código que recibe cada bloque de código Hex del firmware por TFTP y ejecuta su grabación.

Tabla 4.2. Código encargado de recibir cada bloque hex del firmware mediante TFTP y ordenar su programación en la memoria FLASH del microcontrolador.

```
// Decodifica linea de firmware
DecodeHex(NULL, NULL, NULL, NULL, TRUE);

// Get block number
wvBlockNumber.v[1] = *AppDataPtr++;
wvBlockNumber.v[0] = *AppDataPtr++;
AppDataLen -= 2;

// Calcula tamaño del paquete
wvPacketLen.Val = 0 + 4;
wNeededBlock++;

// Perform the decode and flash write
while(AppDataLen){
       static DWORD dwAddress;
       WORD w;
       CHAR cRet;
       w = AppDataLen;
       cRet = DecodeHex(AppDataPtr, &w, vFlashData, &dwAddress, FALSE);

       // Envía bloque válido a grabación
       if(bNeedUpdatedJumpTable){
              if(!RawWriteFlashBlock((DWORD),NULL, NULL)){
                     goto ValidationError;
              }
       }
       SendACK((void*)&vTXBuffer[0]);
}

// Actualizar dirección de escritura
EWRPT = ETXST + 1;
PutMACHeader ((void*)&vTXBuffer[0], (void*)ETHHeaderPtr-
>MACHeader.SourceMAC);
PutIPHeader ((void*)&vTXBuffer[0]);
PutUDPHeader (wvPacketLen);

// Enviar Trama de respuesta TFTP
MACPutArray(vTXBuffer, sizeof(MAC_HEADER) + sizeof(IP_HEADER) +
sizeof(UDP_HEADER));
ETXND = TXSTART + sizeof(MAC_HEADER) + sizeof(IP_HEADER) +
wvPacketLen.Val;
MACFlush();
```

4.1.1.2 Servidor de interacción

La interacción con este laboratorio queda limitada al control de 8 entradas digitales. Por otro lado, al tratarse de un laboratorio cuya principal característica es el bajo coste y la fácil desplegabilidad, la monitorización del laboratorio se lleva a cabo

representando el estado de 10 salidas digítales por medio de 10 leds virtuales mostrados en interfaz gráfico de usuario. La implementación de estas dos capacidades: interacción y monitorización han sido implementadas independientemente para en un futuro poder sustituir una funcionalidad sin influir en la otra (p.e.: inclusión de una cámara IP para el desarrollo de la monitorización).

4.1.1.2.1 Servicio de interacción

La implementación de los comandos de interacción, de acuerdo a la arquitectura planteada, se lleva a cabo mediante un servidor HTTP por medio de comandos GET. La pila TCP/IP proporcionada por Microchip dispone de una función que permite la gestión de comandos HTTP (Tabla 4.3). Mediante dicha función se accede a los datos recibidos en la conexión HTTP actual, discriminando el nombre de la página solicitada y el resto de la información que acompaña en la URL. Únicamente se permite alterar el valor de las salidas desde la página que permite la interacción/monitorización de la plataforma de experimentación (interact.htm).

Tabla 4.3. Función de gestión de comandos HTTP GET utilizada en la implementación del laboratorio remoto WebLab-PIC.

```
HTTP_IO_RESULT HTTPExecuteGet(void) {
	BYTE *ptr;
	BYTE filename[20];

	//Accedemos al nombre de la página que genera la petición HTTP
	MPFSGetFilename(curHTTP.file, filename, 20);

	//Restricción de gestión de interacción desde página interact.htm
	if(!memcmppgm2ram(filename, "interact.htm", 12))
	{
		// Respuesta a comando SwitchON (activar salida)
		ptr = HTTPGetROMArg(curHTTP.data, (ROM BYTE *)"SwitchON");
		if (ptr)
			switch (*ptr) {
				case '0' :	SW0_OUT = 1;
						break;
				case '1' :	SW1_OUT = 1;
						break;
				case '2' :	SW2_OUT = 1;
						break;
				case '3' :	SW3_OUT = 1;
						break;
				case '4' :	SW4_OUT = 1;
						break;
				case '5' :	SW5_OUT = 1;
						break;
				case '6' :	SW6_OUT = 1;
						break;
				case '7' :	SW7_OUT = 1;
						break;
```

```
                        case '8' :   SW8_OUT = 1;
                                     break;
                        case '9' :   SW9_OUT = 1;
                        default  :   break;
                }
            // Respuesta a comando SwitchOFF (desactivar salida)

            ptr = HTTPGetROMArg(curHTTP.data, (ROM BYTE *)"SwitchOFF");
            if (ptr)
                  switch (*ptr) {
                        case '0' :   SW0_OUT = 0;
                                     break;
                        case '1' :   SW1_OUT = 0;
                                     break;
                        case '2' :   SW2_OUT = 0;
                                     break;
                        case '3' :   SW3_OUT = 0;
                                     break;
                        case '4' :   SW4_OUT = 0;
                                     break;
                        case '5' :   SW5_OUT = 0;
                                     break;
                        case '6' :   SW6_OUT = 0;
                                     break;
                        case '7' :   SW7_OUT = 0;
                                     break;
                        case '8' :   SW8_OUT = 0;
                                     break;
                        case '9' :   SW9_OUT = 0;
                        default  :   break;
                  }
      }
      return HTTP_IO_DONE;
}
```

4.1.1.2.2 Servicio de monitorización

La monitorización de las salidas digitales de la plataforma de experimentación se lleva a cabo mediante 8 diodos led virtuales cuyo estado se representa en la página Web en la que se desarrolla la experimentación. La implementación de este componente se ha llevado a cabo utilizando un recurso incluido en la pila TCP/IP de Microchip denominado "variables dinámicas básicas". Básicamente este recurso permite modificar dinámicamente el valor de variables introducidas en las páginas web que son servidas por el servidor HTTP implementado dentro de la pila TCP/IP. Cuando el servidor HTTP recibe la petición desde un cliente, antes de devolver el código HTML correspondiente a la página solicitada, localiza todas las variables dinámicas básicas, sustituyendo la aparición de las mismas en el documento HTML por el resultado de la ejecución de una función asociada a cada variable.

La inclusión de las variables dinámicas básicas en el documento HTML se lleva a cabo colocando el identificador de las mismas acotado por el carácter "~". Cuando, al servir una página web, el servidor localiza una variable dinámica básica automáticamente lanza la ejecución de una función asociada que puede añadir a la trama HTTP hasta 16 caracteres. Esta función no puede recibir ningún parámetro ni lleva a cabo la devolución de ningún valor, sin embargo mediante la función "TCPPutString" permite añadir una cadena de caracteres al contenido de la respuesta del servidor.

Para la implementación del sistema de monitorización, se han definido 16 funciones idénticas que simplemente escriben en la respuesta HTTP la cadena "on" u "off" en función del valor de la entrada o salida digital correspondiente del microcontrolador en el que se ha implementado el servidor de interacción. Estas funciones permiten monitorizar el valor de las señales de la plataforma de experimentación en el interfaz gráfico de usuario. La Tabla 4.4 muestra la función correspondiente al led 0 que se monitorizará en el interfaz gráfico de usuario implementado en el cliente.

Tabla 4.4. Función HTTPPrint_led0 que reemplaza la variable dinámica básica ~led0~ por la cadena de caracteres "on" u "off" en función del valor que se recibe de a través del puerto PORTE0 del microcontrolador PIC18F97J60 sobre el que se implementa el servidor de interacción del WebLab-PIC.

```
void HTTPPrint_led0(void)
{
	if (LED0_IO)
		TCPPutString (sktHTTP, "on");
	else
		TCPPutString (sktHTTP, "off");

}
```

4.1.1.3 Plataforma de experimentación

Como se puede observar en la Figura 4.3, la plataforma de experimentación consiste en un microcontrolador PIC18F97J60 cuya programación se lleva a cabo mediante el bootloader descrito en el apartado 4.1.1.1. La implementación del bootloader permite al estudiante hacer uso de todos los recursos del microcontrolador con la única limitación de los últimos 4KBytes de la memoria de programa, utilizados para albergar el código del bootloader. La plataforma de experimentación dispone de 8 entradas digitales que pueden ser controladas durante la validación del experimento y 8 salidas digitales que pueden ser monitorizadas. La Tabla 4.5 muestra los recursos hardware dispuestos para la interacción/monitorización del experimento.

Tabla 4.5. Definición de las entradas y salidas digitales utilizadas para la interacción con el laboratorio remoto WebLab-PIC.

```
// Macros para el control de las salidas
#define SW0_TRIS                (TRISBbits.TRISB0)
#define SW0_OUT                 (LATBbits.LATB0)
#define SW1_TRIS                (TRISBbits.TRISB1)
#define SW1_OUT                 (LATBbits.LATB1)
#define SW2_TRIS                (TRISBbits.TRISB2)
#define SW2_OUT                 (LATBbits.LATB2)
#define SW3_TRIS                (TRISBbits.TRISB3)
#define SW3_OUT                 (LATBbits.LATB3)
#define SW4_TRIS                (TRISBbits.TRISB4)
#define SW4_OUT                 (LATBbits.LATB4)
#define SW5_TRIS                (TRISBbits.TRISB5)
#define SW5_OUT                 (LATBbits.LATB5)
#define SW6_TRIS                (TRISBbits.TRISB6)
#define SW6_OUT                 (LATBbits.LATB6)
#define SW7_TRIS                (TRISBbits.TRISB7)
#define SW7_OUT                 (LATBbits.LATB7)

// Macros para la monitorización  de las entradas
#define LED0_TRIS               (TRISDbits.TRISD0)
#define LED0_IO                 (LATDbits.LATD0)
#define LED1_TRIS               (TRISDbits.TRISD1)
#define LED1_IO                 (LATDbits.LATD1)
#define LED2_TRIS               (TRISDbits.TRISD2)
#define LED2_IO                 (LATDbits.LATD2)
#define LED3_TRIS               (TRISDbits.TRISD3)
#define LED3_IO                 (LATDbits.LATD3)
#define LED4_TRIS               (TRISDbits.TRISD4)
#define LED4_IO                 (LATDbits.LATD4)
#define LED5_TRIS               (TRISDbits.TRISD5)
#define LED5_IO                 (LATDbits.LATD5)
#define LED6_TRIS               (TRISDbits.TRISD6)
#define LED6_IO                 (LATDbits.LATD6)
#define LED7_TRIS               (TRISDbits.TRISD7)
#define LED7_IO                 (LATDbits.LATD7)
```

4.1.1.4 Servidor del laboratorio

La independencia funcional del servidor de interacción y del servidor de grabación simplifica el desarrollo del servidor del laboratorio del WebLab-PIC que únicamente desarrolla una función: Recibir el firmware desarrollado por el estudiante utilizando un determinado entorno de desarrollo y enviarlo convenientemente al servidor de grabación.

En primer lugar, el servidor del laboratorio recibe, mediante un comando HTTP tipo POST, el fichero binario en formato Intel HEX de 8 bits que contiene el firmware desarrollado por el estudiante. La limitación de memoria RAM del microcontrolador en el que se implementa el servidor de laboratorio, con menos de 1Kbyte libre, exige que el tratamiento del firmware se ejecute línea a línea según estas son recibidas.

Para cada línea del firmware, el servidor del laboratorio comprueba la validez de la misma comprobando que la dirección de escritura de los datos en la memoria de programa no excede la memoria del microcontrolador ni sobre-escribe el código del bootloader (1DC00h – 1FB08h). En caso de que una línea no cumpla el formato Intel HEX, o que la dirección de escritura se encuentre fuera del rango establecido, el servidor de laboratorio responde con un error 404 indicando la invalidez del código enviado.

Tabla 4.6. Función principal del servidor del laboratorio del laboratorio remoto WebLab-PIC que recibe mediante un comando HTTP POST un fichero con formato Intel HEX y lo envía al bootloader implementado como servidor de grabación.

```
while (curHTTP.smPost != SM_FW_POST_COMPLETE)
{
	//Comprueba que una línea completa se ha recibido
	lenA = TCPFindROMArray(sktHTTP, (ROM BYTE*)"\r\n", 2, 0, FALSE);
	if(lenA == 0xffff)
	{//En caso contrario pide más datos
		return HTTP_IO_NEED_DATA;
	}

	// Si se ha recibido la línea completa actualiza longitud de línea
	curHTTP.byteCount -= lenA + 6;

	// Parsea la última línea recibida
	curHTTP.byteCount -= TCPGetArray(sktHTTP, NULL, lenA+2);

	if (!TestCheckSum(sktHTTP, curHTTP.data, curHTTP.byteCount))
	{
		return HTTP_FILE_CS_ERROR
	}

	if (!UploadLine(sktHTTP, curHTTP.data, curHTTP.byteCount))
	{
		return HTTP_FILE_FORMAT_ERROR
	}

	// Si no hay más líneas de datos
	if(curHTTP.byteCount == 0u)
	{
		curHTTP.smPost = SM_FW_POST_COMPLETE
		return HTTP_IO_DONE;
	}

	// en caso contrario espera a siguiente línea
	return HTTP_IO_NEED_DATA;
}

return HTTP_IO_DONE;
```

4.1.1.5 Cliente

El cliente del laboratorio remoto WebLab-Pic proporciona el interfaz gráfico de usuario que permite el envío del firmware y la interacción/monitorización de las entradas y salidas digitales de la plataforma de experimentación. El cliente está desarrollado en lenguaje HTML, utilizando AJAX para la continua actualización del módulo de monitorización que representa el valor de los leds.

La web se encuentra almacenada dentro del sistema de ficheros MPFS (Microchip File System) implementado sobre la memoria de programa del microcontrolador PIC18F97J60 que alberga este componente del laboratorio remoto. Incluye información del uso del laboratorio y ejemplos de programas en los que el estudiante puede basar su experimentación.

El estudiante con acceso al laboratorio puede alternar libremente entre las páginas que permiten la subida del firmware y la interacción con el experimento. La página de subida del firmware contiene un simple formulario que permite la búsqueda de archivos Intel HEX en los sistemas de ficheros del dispositivo cliente y envía el fichero seleccionado al mismo servidor a través de un comando POST, para que el servidor del laboratorio se encargue de transmitir el firmware al bootloader vía TFTP a través del puerto 69.

Por su parte la página de interacción (Figura 4.9) permite monitorizar el estado de las salidas de la plataforma de interacción y alterar el valor introducido a través de las entradas. Las salidas están contenidas en un contenedor que se actualiza 5 veces por segundo por medio de una función en JavaScript, permitiendo monitorizar los cambios con una mínima latencia de 200ms. Cada salida está representada por una imagen que representa una bombilla encendida (Salida=1) o apagada (Salida=0). El nombre de los ficheros que contienen ambas imágenes es respectivamente "ledon.png" y "ledoff.png" de tal forma que las variables dinámicas básicas implementadas en el servidor de interacción se encargan de indicar el estado de cada salida (Tabla 4.7).

Respecto a la implementación de las 8 entradas que permiten la interacción, cuatro han sido implementadas como interruptores y cuatro como pulsadores que generan un pulso alto de longitud variable en base a un campo de texto. La emulación de los interruptores se ha desarrollado mediante variables dinámicas básicas que alteran el URL del enlace dinámico asociado a cada imagen que representa un interruptor cerrado o abierto, llamando al método SwitchOFF o SwitchON respectivamente. El pulso generado por los pulsadores virtualizados se provoca desde una función en JavaScript que ejecuta los métodos SwitchON y SwitchOFF,

dejando un intervalo entre ambos definido por el estudiante mediante un campo de texto (Figura 4.9).

Tabla 4.7. Contenedor que permite la monitorización de las salidas del laboratorio remoto WebLab-Pic en el interfaz gráfico de usuario en funcion de las variables dinámicas básicas ~led0~ a ~led7~.

```
<DIV ID="online">
      <TABLE CLASS="TABLA">
            <TR><TD CLASS="TITULO">RD0</TD>
                  <TD><img src="images/led~led0~.png"></TD></TR>
            <TR><TD CLASS="TITULO">RD1</TD>
                  <TD><img src="images/led~led1~.png"></TD></TR>
            <TR><TD CLASS="TITULO">RD2</TD>
                  <TD><img src="images/led~led2~.png"></TD></TR>
            <TR><TD CLASS="TITULO">RD3</TD>
                  <TD><img src="images/led~led3~.png"></TD></TR>
            <TR><TD CLASS="TITULO">RD4</TD>
                  <TD><img src="images/led~led4~.png"></TD></TR>
            <TR><TD CLASS="TITULO">RD5</TD>
                  <TD><img src="images/led~led5~.png"></TD></TR>
            <TR><TD CLASS="TITULO">RD6</TD>
                  <TD><img src="images/led~led6~.png"></TD></TR>
            <TR><TD CLASS="TITULO">RD7</TD>
                  <TD><img src="images/led~led7~.png"></TD></TR>
      </TABLE>
</DIV>
```

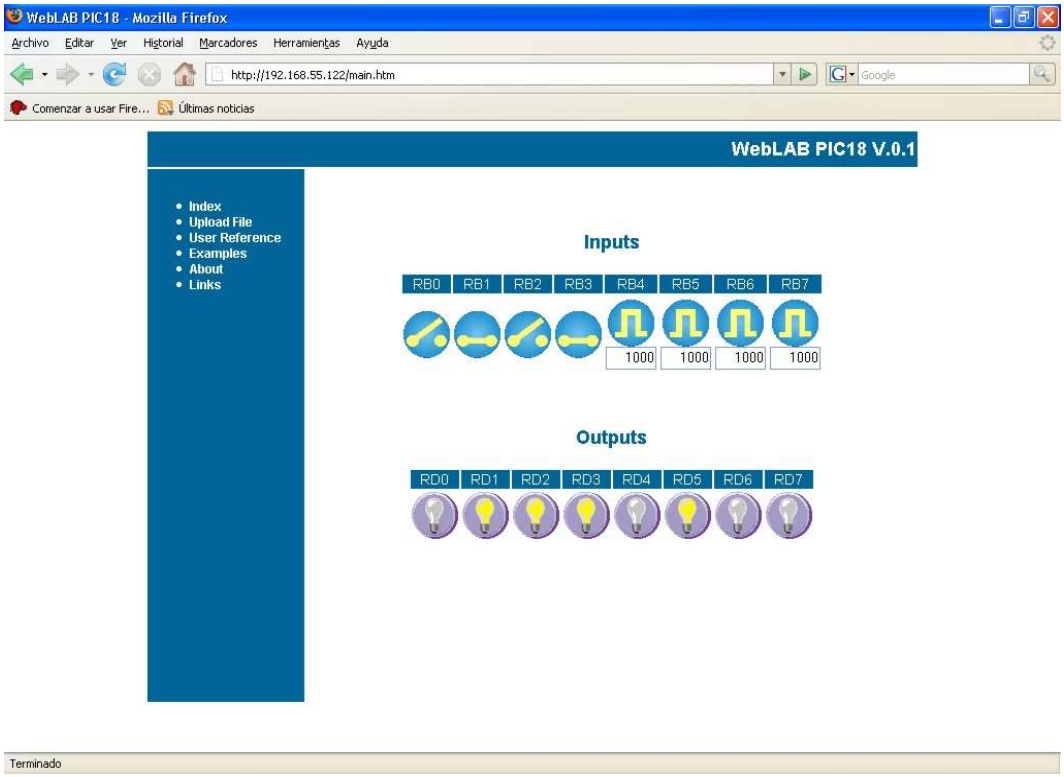

Figura 4.9. Interfaz gráfico de usuario del laboratorio remoto WebLab-PIC.

4.1.1.6 Servidor de administración

Sin duda el gran perjudicado de implementar todos los componentes del laboratorio a excepción del servidor de grabación y de la plataforma de experimentación sobre un microcontrolador de 8 bits con recursos limitados es el servidor de administración. El consumo del 90% de la memoria de programa del microcontrolador para la implementación del servidor del laboratorio, del servidor de interacción y del cliente forzó a desarrollar un servidor de administración con la funcionalidad mínima para soportar el correcto desempeño de la experimentación en el laboratorio remoto.

Concretamente las funciones que desempeña el servidor de administración son las siguientes:

- ***Autenticación:*** el requisito establecido para el desarrollo de este laboratorio remoto de soportar la experimentación con microcontroladores PIC de forma independiente implica desarrollar un sistema propio de autenticación. El limitado espacio restante en memoria FLASH impide el almacenamiento de las credenciales de los estudiantes en este recurso. Afortunadamente el sistema de desarrollo utilizado para la implementación del laboratorio, la tarjeta PICDEMNET2 de Microchip (Microchip Technology, 2006), cuenta con una memoria EEPROM 24LC256 conectada al microcontrolador mediante bus SPI (Serial Periferal Interface) que proporciona 256Kbits que serán utilizados con este propósito. Para cada usuario simplemente se almacena un identificador y una contraseña con una longitud máxima de 8 caracteres ASCII para cada dato. Esto permite almacenar hasta un total de 2048 credenciales. La inserción, modificación y borrado de los usuarios se lleva a cabo mediante una web que utiliza el método de autenticación de la pila TCPIP proporcionada por Microchip.

- ***Gestión de sesiones:*** Una vez el usuario se autentica aparecen una lista de todas las instancias de experimentación disponibles indicando si están reservadas o no. La búsqueda de instancias en la red se lleva a cabo mediante el protocolo TCPIP Discoverer v1.1, proporcionado por la pila TCPIP de Microchip para la búsqueda en la red de dispositivos que la implementan. Este protocolo devuelve las direcciones IP y MAC de todos los dispositivos conectados a la red que implementan este protocolo sobre el puerto 30303. Cuando un usuario autenticado accede a una instancia libre, esta se marca como reservada y únicamente es accesible desde la dirección MAC de ese usuario. El tiempo de cada sesión es de 120 segundos, tras los cuales el cliente devuelve al usuario a la lista de instancias detectadas. La limitación de los recursos disponibles impide desarrollar un sistema de gestión de colas o reservas, pero el bajo coste del laboratorio permite disponer de un número de

instancias que se ajuste al dominio de usuarios facilitando el acceso de estos. Poder aumentar el número adecuado de instancias de experimentación según el dominio de estudiantes, aporta un grado óptimo de escalabilidad al laboratorio.

4.1.2 Experiencia de usuario del laboratorio remoto WebLab-PIC

Pese a las limitaciones identificadas durante la implementación, este laboratorio remoto permite al estudiante desarrollar experimentación real con microcontroladores PIC de manera análoga a como se desempeña en un laboratorio presencial.

Puesto que el bootloader encargado de la programación no requiere ninguna modificación en el firmware desarrollado por el estudiante, este laboratorio permite la utilización del entorno de desarrollo software preferido por el estudiante. El estudiante debe desarrollar el programa y generar el firmware resultante utilizando cualquier herramienta externa para el desarrollo de sistemas basados en microcontroladores PIC (MPLAB IDE, MPLAB X, CCS PICC, EasyPIC, etc.).

Una vez dispone de un firmware codificado el estudiante autorizado debe autenticarse con sus credenciales para acceder a una página que permite la elección de la instancia sobre la que desea desarrollar la experimentación (Figura 4.10). Si la instancia se encuentra disponible el alumno podrá iniciar su sesión y dispondrá de 120 segundos para probar el firmware desarrollado.

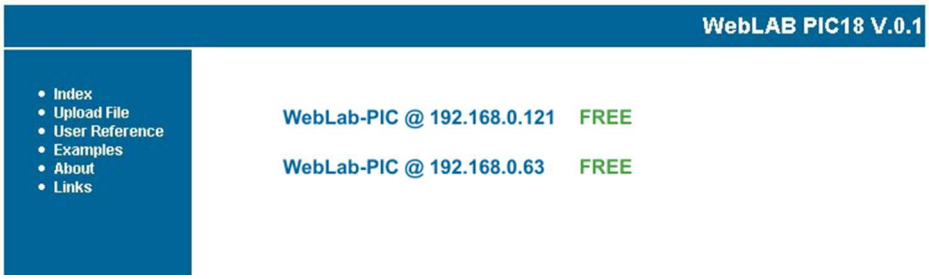

Figura 4.10. Interfaz para la selección de la instancia de experimentación en el laboratorio remoto WebLab-PIC.

El primer paso para llevar a cabo la experimentación consisten en programar el dispositivo con el firmware previamente desarrollado. El alumno debe seleccionar el fichero binario generado con formato Intel HEX y subirlo a la plataforma de experimentación (Figura 4.11).

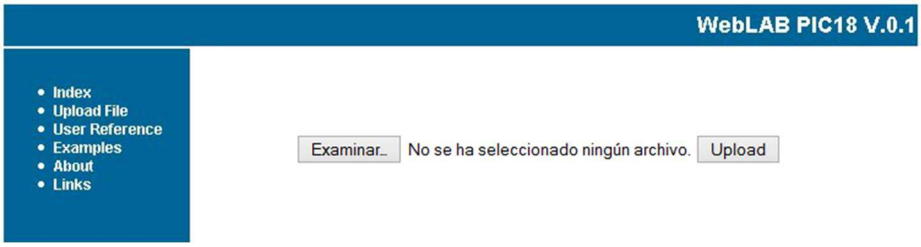

Figura 4.11. Interfaz para la selección del firmware sobre el que desarrollar la experimentación en el laboratorio remoto WebLab-PIC.

Una vez el PIC18F97J60 sobre el que se desarrolla la experimentación es programado con el experimento, el alumno dispondrá de un interfaz gráfico de usuario desde el cual podrá monitorizar las salidas generadas por el sistema y controlar las entradas por medio de interruptores y pulsadores virtuales (Figura 4.9).

4.1.3 Resumen del laboratorio remoto WebLab-PIC

El laboratorio WebLab-PIC representa una implementación mínima de la arquitectura abierta para el despliegue de laboratorios remotos con experimentación en sistemas embebidos. Todos los componentes identificados en la arquitectura han sido implementados de acuerdo a las especificaciones establecidas de funcionalidad independiente. Este laboratorio remoto soporta la experimentación con el microcontrolador PIC 18F97J60 (Figura 4.12) y permite al estudiante llevar a cabo la programación del dispositivo y la validación del firmware mediante la interacción con entradas y salidas digitales.

Figura 4.12. Laboratorio remoto WebLab-PIC con dos instancias de experimentación.

Su implementación ha sido llevada a cabo pensando persiguiendo su fácil desplegabilidad. Actualmente, únicamente requiere alimentación y una toma de red para su despliegue en cualquier infraestructura de red con DHCP (protocolo de configuración dinámica de host) y ya es accesible por los estudiantes de la misma. Lógicamente para permitir la disponibilidad desde el exterior de la red donde este laboratorio remoto es desplegado es necesario que los servicios informáticos configuren el acceso.

4.2 Laboratorio remoto WebLab-Robot

El laboratorio remoto WebLab-Robot permite la experimentación con microcontroladores de 8 bits sobre un robot móvil capaz de detectar obstáculos en su frontal y distinguir el color de la superficie por la que circula (Dziabenko, Garcia-Zubia, & Angulo, 2012).

Las ventajas de utilizar robots móviles para el aprendizaje de microcontroladores en los primeros cursos de ingeniería (Jung, 2013) (Mondada, y otros, 2009) motivaron el desarrollo de este laboratorio remoto como un recursos adicional a emplear en la asignatura de Microcontroladores, cursada en el segundo cuatrimestre del segundo curso de Ingeniería Técnica en Automática y Electrónica Industrial.

Paralelamente este laboratorio remoto fue elegido dentro del proyecto e-Pragmatic (Rojko, Jezernik, & Pester, 2011) como herramienta educativa para fomentar la formación continua en microcontroladores. Desde el día en que fue lanzado en producción a través de la plataforma WebLab-Deusto este laboratorio acumula más de 7000 sesiones. Instancias de este laboratorio han sido desplegadas en el MIT - Massachusetts Institute of Technology, dentro del proyecto "Building an ecology of Online Laboratories" (Orduña P. , y otros, 2012) subvencionado por la NSF (National Science Foundation) (Award#: 1132813) y en la Universidad Estatal Shota Rustaveli en Batumi, Georgia (Figura 4.13) dentro del proyecto iCo-op subvencionado dentro del Programa Tempus de la Comunidad Europea (0278-TEMPUS-1-2012-1-DE-Tempus-JPH) (Arras, Henke, Tabunshchyk, & Van Merode, 2015).

Si bien el propósito del laboratorio remoto WebLab PIC consistía en demostrar la posibilidad de implementar una versión ligera de la arquitectura planteada, el objetivo principal de la implementación de este laboratorio remoto es evaluar la integración de la arquitectura propuesta en sistemas de gestión de laboratorios remotos. Aunque como se verá a continuación todos los componentes del laboratorio han sido diseñados siguiendo las indicaciones definidas en la arquitectura y aprovechando los recursos que ofrece el RLMS WebLab-Deusto, posteriormente el

laboratorio remoto ha sido también integrado en el sistema de gestión de laboratorios remotos iLab (Harward, y otros, 2008) para garantizar su adecuación a distintas plataformas.

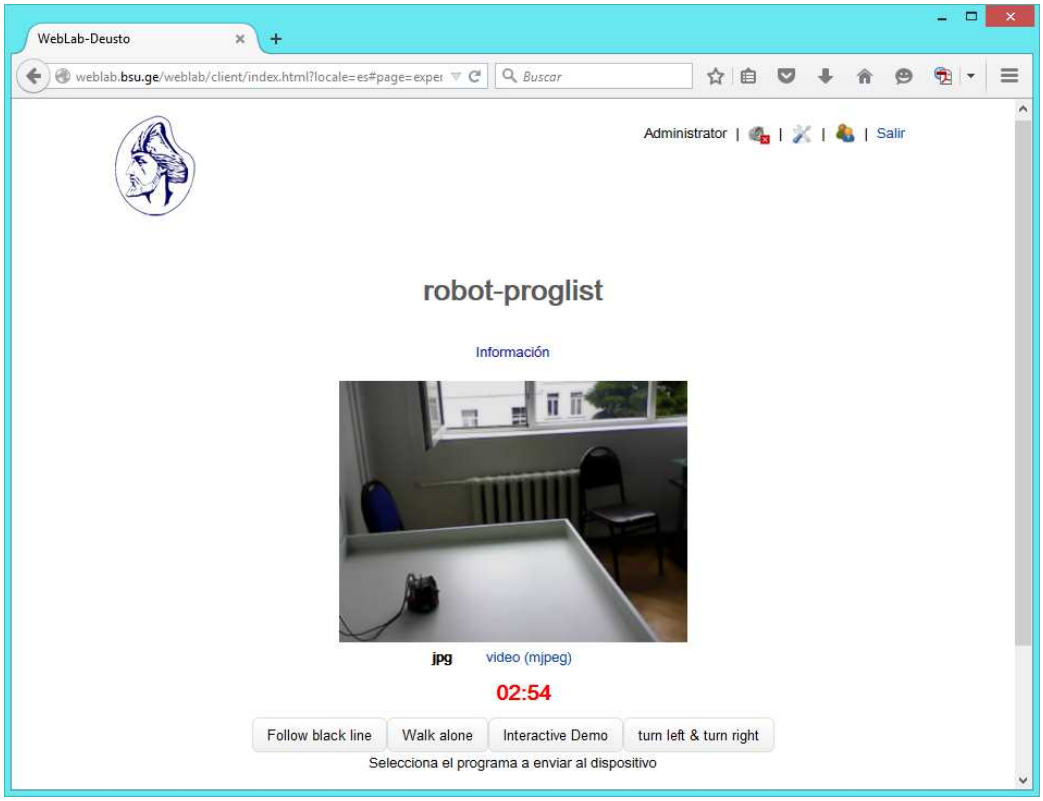

Figura 4.13. Instancia del laboratorio WebLab-Robot desplegada en la Universidad Estatal de Shota Rustabeli en Batumi, Georgia.

Además el despliegue de múltiples instancias del laboratorio en distintas instituciones, garantiza la desplegabilidad del mismo y demuestra la escalabilidad.

4.2.1 Implementación del laboratorio remoto WebLab-Robot

A diferencia del laboratorio remoto WebLab-PIC en el que uno de los requisitos fundamentales consistía en la capacidad para funcionar de manera independiente. El laboratorio remoto WebLab-Robot fue desarrollado para validar la arquitectura propuesta en un laboratorio remoto integrado en el sistema de gestión de laboratorios remotos WebLab-Deusto. Este RLMS proporciona todas las características de administración necesarias para el óptimo consumo del laboratorio por parte del estudiante (Orduña, 2013)(Orduña, y otros, 2011) y además facilita el desarrollo y la integración de los servidores del laboratorio y cliente proporcionando una API

(Application Programming Interface) con las principales funciones requeridas para el desarrollo de un laboratorio remoto.

A continuación se detallan los principales componentes de este laboratorio remoto, de acuerdo a la arquitectura propuesta.

4.2.1.1 Servidor de grabación

Cumpliendo con la arquitectura propuesta y dado que la plataforma de experimentación está formada por un robot móvil, el servidor de grabación debe estar implementado mediante una solución independiente, pero además libre de cables para no limitar la movilidad de la plataforma de experimentación. La disponibilidad del microcontrolador PIC18F4520, sobre el que se desarrolla la experimentación, de un puerto USART (Universal Synchronous Asynchronous Receiver Transmiter) así como la capacidad de reprogramación de la memoria de programa en tiempo de ejecución, posibilita el desarrollo de un bootloader capaz de realizar la reprogramación del microcontrolador.

En base al bootloader proporcionado por Microchip en su nota de aplicación AN851 (Microchip Technology, 2002) para la reprogramación de microcontroladores a través del protocolo RS232, se desarrollaron las modificaciones necesarias para minimizar el espacio de la memoria de programa utilizado y permitir la programación del microcontrolador a través de tecnología bluetooth, mediante un adaptador bluetooth a serie (A7 engineering, 2009).

Para minimizar el tamaño del código del bootloader la implementación del mismo se ha llevado a cabo mediante lenguaje ensamblador (Tabla 4.8) ocupando menos de 200 instrucciones (Figura 4.14).

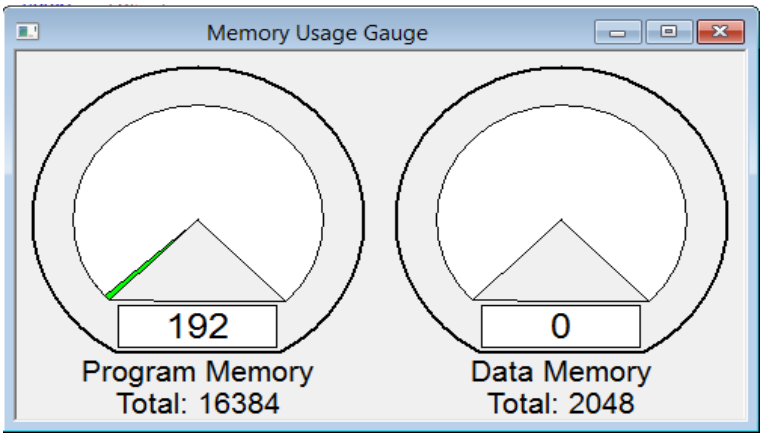

Figura 4.14. Recursos utilizados por el bootloader sobre el que se implementa el servidor de grabación del laboratorio remoto WebLab-robot.

Tabla 4.8. Código fundamental del bootloader del servidor de grabación del laboratorio remoto WebLab-Robot que lleva a cabo la reprogramación del código recibido.

```
WriteProgMem
      movlw  b'11111000'         ; Posicionarse a inicio de bloque
      andwf  TBLPTRL, F
      movlw  .16

Lp1   movff  POSTINC0, TABLAT    ; Carga el valor del bloque en TABLAT
      tblwt *+                   ; y post-incrementa
      decfsz WREG, F             ; Decrementa contador
      bra    Lp1                 ; Hasta completar bloque

      tblrd  *-                  ; Reposiciona TBLPTR en el bloque

      movlw  b'10000100'         ; Configura memoria FLASH y Write Enable
      movwf  EECON1
      movlw  0x55
      movwf  EECON2
      movlw  0xAA                ; Activa Trigger EECON2
      movwf  EECON2
      bsf    EECON1, WR          ; Ejecuta grabación del blqoue
      nop                        ; Espera 1 ciclo para grabación

      tblrd  *+                  ; apuntar al próximo bloque
      decfsz COUNTER, F
      bra    WriteProgMem        ; Repetir para siguiente bloque

      bra    sendACK             ;  Si fin, Enviar ACK
```

La utilización de este bootloader implica reubicar el vector de reset y los vectores de interrupciones modificando sus direcciones como se observa en la Tabla 4.9.

Tabla 4.9. Redefinición de vectores en el laboratorio remoto WebLab-Robot.

	PIC18F4550	WebLab-Robot
Vector de reset	0h	200h
Vector de int. de alta prioridad	08h	208h
Vector de int. de baja prioridad	18h	218h

Desde el mismo laboratorio remoto los estudiantes pueden descargar los ficheros de lincaje para los compiladores MPLAB-C18 y XC8 que permiten el desarrollo de los experimentos sin tener en cuenta esta alteración en el posicionamiento de los vectores.

4.2.1.2 Servidor de Interacción

Al tratarse de un laboratorio remoto para experimentación con un robot móvil, una de las principales características que deben cumplir los experimentos desarrollados por los estudiantes es permitir el funcionamiento autónomo del robot móvil. Por tanto, en el modo estándar, este laboratorio remoto carece de capacidad de interacción ya que todas las entradas del sistema corresponden a la propia sensórica del robot móvil.

Sin embargo, en ocasiones es necesario permitir al estudiante recolocar al robot móvil en una posición concreta dentro de la pista sobre la que se desplaza el robot móvil, para la validación de los bloque individuales del programa que resuelven un determinado comportamiento (p.e.: colocar el robot sobre la línea negra). Para cubrir esta necesidad se ha desarrollado un modo de interacción que permite al estudiante desplazar el robot mediante un joystick virtual dispuesto en el interfaz gráfico de usuario (Figura 4.15).

Figura 4.15. Interfaz de usuario para desplazar libremente el robot sobre la pista del laboratorio remoto WebLab-Robot.

Debido a que el control de los motores está directamente controlado por el microcontrolador sobre el que se realiza la experimentación, es necesario programar dicho microcontrolador con un programa que reciba los comandos de movimiento mediante el módulo bluetooth y los ejecute sobre los motores. Para evitar que el estudiante pueda dañar al robot, el programa generado utiliza los sensores de obstáculos instalados en el robot para evitar las paredes de la pista (Tabla 4.10).

Tabla 4.10. Rutina de servicio de interrupción que ejecuta las órdenes recibidas desde el módulo bluetooth en el laboratorio remoto WebLab-Robot.

```
Inter  movf   RCREG,W
       sublw  'F'
       btfsc  STATUS,Z
       bra    Forward ; Comando “F”
       movf   RCREG,W
       sublw  'L'
       btfsc  STATUS,Z
       bra    Left    ; Comando “L”
       movf   RCREG,W
       sublw  'B'
       btfsc  STATUS,Z
       bra    Back    ; Comando “B”
       movf   RCREG,W
       sublw  'R'
       btfsc  STATUS,Z
       bra    Right  ; Comando “R”
       bcf    PIR1,RCIF ; Otro Comando, Reset Flag y vuelta a main
       retfie

Forward      bsf    PORTC,1
             bcf    PORTC,0            ;Motor 1 PARADO
             bcf    PORTD,3
             bsf    PORTC,2            ;motor 2 PARADO
             bra    Espera1s

Back         bcf    PORTC,1
             bsf    PORTC,0            ;Motor 1 PARADO
             bsf    PORTD,3
             bcf    PORTC,2            ;motor 2 PARADO
             bra    Espera1s

Left         bsf    PORTC,1
             bcf    PORTC,0            ;Motor 1 PARADO
             bsf    PORTD,3
             bcf    PORTC,2            ;motor 2 PARADO
             bra    Espera025s

Right        bcf    PORTC,1
             bsf    PORTC,0            ;Motor 1 PARADO
             bcf    PORTD,3
             bsf    PORTC,2            ;motor 2 PARADO
             bra    Espera025s

IntReturn    bcf    PORTC,1
             bcf    PORTC,0            ;Motor 1 PARADO
```

```
            bcf    PORTD,3
            bcf    PORTC,2              ;motor 2 PARADO
            bcf    PIR1,RCIF

            retfie
```

4.2.1.3 Plataforma de experimentación

La plataforma de experimentación se fundamenta en el robot didáctico comercial Azkar-Bot (Ingeniería de Microsistemas Programados, 2010) comercializado por Microsystems Engineering. Este robot se fundamenta en el sistema de desarrollo PIC'Control (Ingeniería de Microsistemas Programados, 2008) para experimentación con el microcontrolador de propósito general PIC18F4550 de Microchip (Microchip Technolgy, 2009)(Figura 4.16).

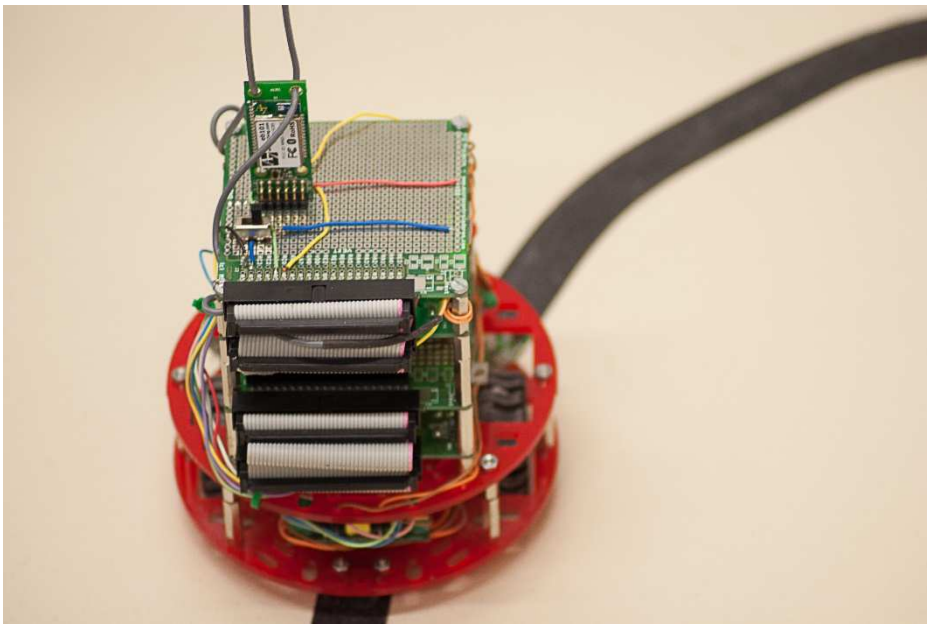

Figura 4.16. Plataforma e experimentación del laboratorio remoto WebLab-Robot.

Debido a que la plataforma de experimentación debe ser reseteada al término de cada sesión de los estudiantes, se ha optado por la duplicación del sistema de desarrollo, incluyendo un microcontrolador "supervisor" que se encarga de monitorizar las sesiones y establecer la comunicación con el servidor del laboratorio y otro dispositivo idéntico "controlador del experimento" cuya única función es el desarrollo de los experimentos de los estudiantes para el control del robot móvil. Esto permite al alumno desarrollar los mismos experimentos a través del laboratorio remoto que los realizados sobre la plataforma física, manteniendo siempre el control de la plataforma de experimentación desde el servidor del laboratorio.

La Figura 4.17 muestra como los dos sistemas de desarrollo integrados en la plataforma de experimentación del laboratorio remoto WebLab-Robot reciben la información enviada desde el servidor del laboratorio a través del módulo bluetooth. El supervisor interpreta los comandos recibidos y notifica el resultado de su ejecución, mientras que el controlador del experimento únicamente espera hasta la recepción el código del firmware que debe reprogramar en su memoria de programa al inicio de cada sesión. Cada vez que el supervisor recibe un comando de "fin de sesión" provoca un reset en el controlador del experimento, mediante la activación del pin MCLR' a través de una GPIO, reiniciando el bootloader para quedar a la espera de un nuevo firmware.

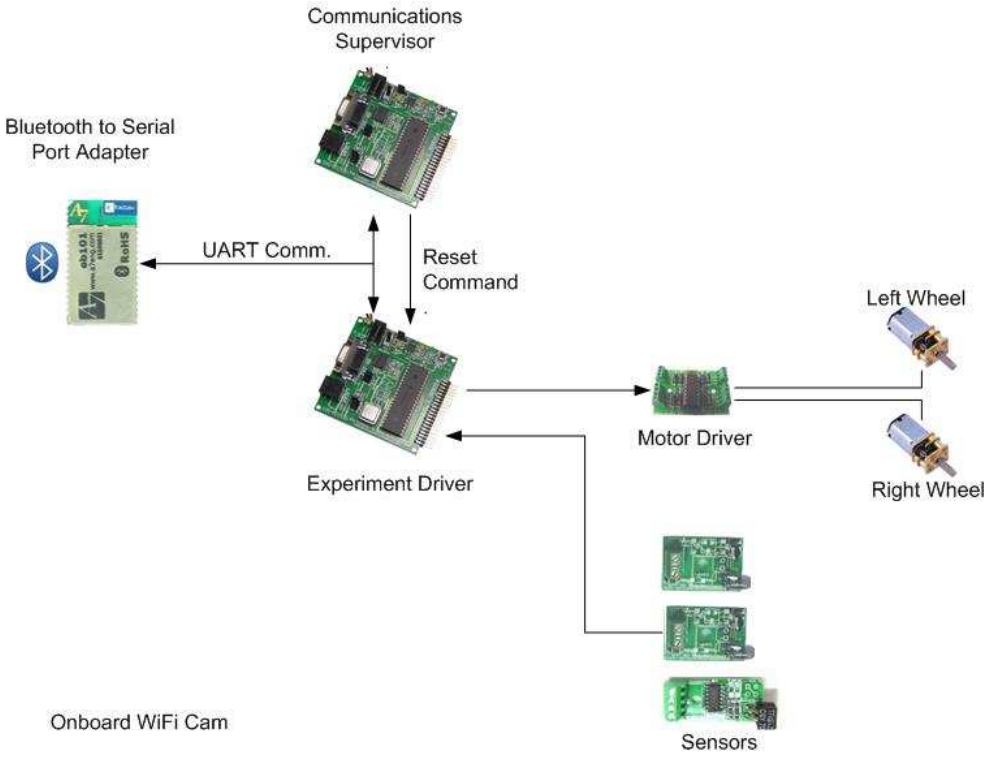

Figura 4.17. Descomposición modular de la plataforma de experimentación del laboratorio remoto WebLab-Robot.

Los recursos del microcontrolador PIC18F4550 que soporta a experimentación reservados para el control de los sensores y actuadores del robot móvil se indican en la Figura 4.18. La elección de los puertos del microcontrolador ha sido escogida para permitir a los estudiantes desarrollar experimentación con los principales recursos del microcontrolador. Los principales elementos que permiten controlar el comportamiento son los siguientes:

- 2 Motores Pololu Micro Metal Gearmotors controlados mediante los pines RC1 y RD3, asociados a sendos módulos de modulación de ancho de banda (PWM) integrados en el microcontrolador
- 2 Sensores de línea (CNY70) conectados a los pines RA0 y RA1 del microcontrolador que pueden comportarse como entradas digitales (blanco o negro) o como canales del convertidor analógico-digital permitiendo detectar la proximidad a la línea negra más cercana.
- 2 Sensores digitales de obstáculos (mse-110) conectados a las entradas digitales RA2 y RA3 que permiten detectar un obstáculo a distancias inferiores a 4 cm.

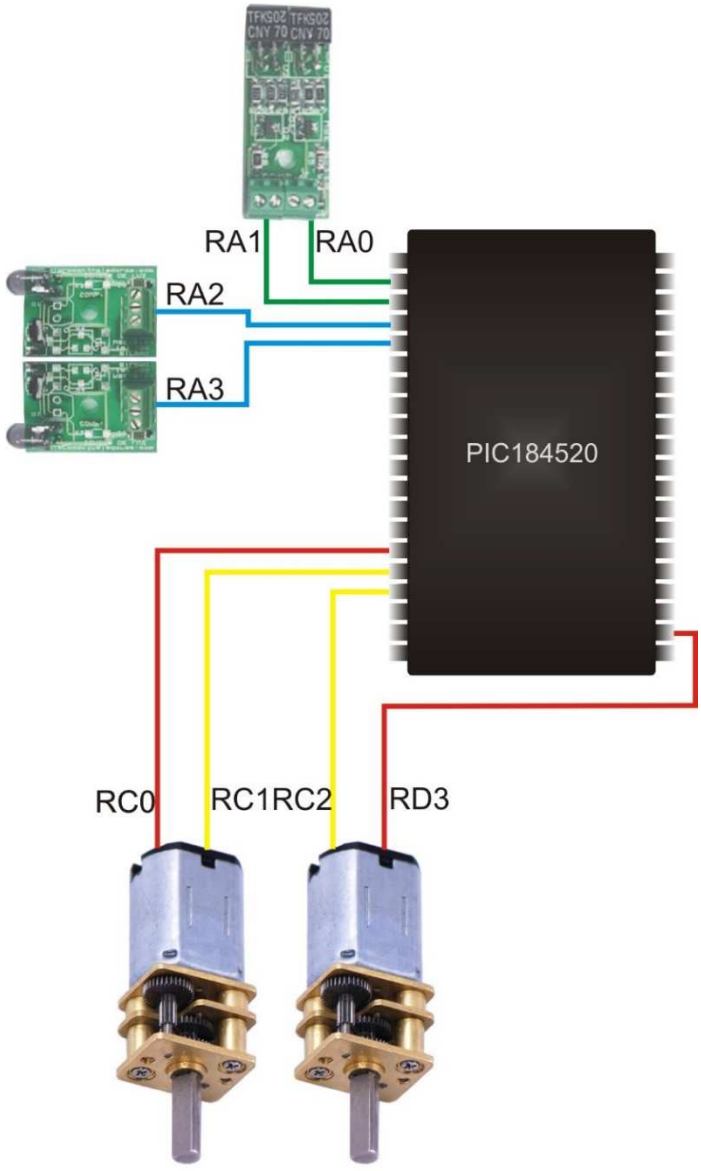

Figura 4.18. Recursos del microcontrolador utilizados para el control del robot móvil en el laboratorio remoto WebLab-Robot.

4.2.1.4 Servidor del laboratorio

La integración del laboratorio remoto WebLab-PIC en el RLMS WebLab-Deusto ha facilitado el desarrollo del servidor del laboratorio. WebLab-Deusto proporciona un método denominado "managed" para el desarrollo del servidor del laboratorio que permite mantener un registro de todas las acciones ejecutadas por el estudiante durante cada sesión de experimentación.

El desarrollo de este laboratorio remoto coincidió con el lanzamiento de la API de WebLab-Deusto para .NET, que fue utilizada para el desarrollo del servidor del laboratorio, sirviendo además como validación de la misma. La Tabla 4.11 muestra la clase que define la comunicación con el RLMS WebLab-Deusto para el desarrollo de las acciones requeridas por el laboratorio remoto WebLab-Robot.

Tabla 4.11. Definición de la clase SampleExperimentServer que recoge las tareas a desarrollar por el servidor del laboratorio del sistema WebLab-Robot.

```
class SampleExperimentServer : WebLabDeusto.ExperimentServer {
    private WebBot20_Pannel.MainPannel panel;
    private System.Collections.Generic.Dictionary<string, string>
descr2fichero = new System.Collections.Generic.Dictionary<string,
string>();
    private System.Collections.Generic.Dictionary<string, string>
fichero2descr = new System.Collections.Generic.Dictionary<string,
string>();

    public SampleExperimentServer(WebBot20_Pannel.MainPannel panel)
    {
        this.panel = panel;
        descr2fichero.Add("Follow white line", "followline.hex");
        descr2fichero.Add("Walk alone", "walkalone.hex");
        foreach(string key in descr2fichero.Keys)
            fichero2descr.Add(descr2fichero[key], key);
    }

        public void StartExperiment()
        {
            Console.WriteLine("Experiment started");
        }

        public string SendFile(byte[] file, string fileInfo)
        {
            // if (enviarFicherosEnabled)
                // return "ERR:Can't send file";
            try
            {
                this.panel.enviarFichero(file);
                return "OK";
            }
            catch (Exception e) {
                return "ERR:" + e.Message;
            }
        }

        public string SendCommand(string command){
            if (command == "programs") {
```

```
                string s = "";
                foreach (string descr in descr2fichero.Keys)
                    s += descr + ",";
                return s;
            }
            else if (command.StartsWith("program:")) {
                string descrName = command.Substring("program:".Length);
                string fileName = descr2fichero[descrName];
            }
            else if (command.StartsWith("move:")) {
                string movement = command.Substring("move:".Length);
            }
            Console.WriteLine("Command received: {0}", command);
            return "Command received: " + command;
        }

        public void Dispose(){
            Console.WriteLine("Experiment disposed");
        }
}
```

Entre las tareas ejecutadas por el servidor del laboratorio, la más compleja es la adecuación del firmware enviado por el estudiante en formato Intel Hex al servidor de grabación implementado sobre un bootloader que recibe las instrucciones en código máquina a través de un módulo bluetooth. La Tabla 4.12 contiene la función del servidor del laboratorio que interpreta cada línea del firmware en formato Intel Hex, enviando las instrucciones que lo componen en modo "big endian" (enviando primero los bytes más significativos) a través del puerto RFCOM sobre la conexión bluetooth establecida con el módulo integrado en la plataforma de experimentación.

Tabla 4.12. Código C# para la interpretación de las líneas del firmware en formato Intel Hex y su adecuación al bootloader bluetooth.

```
While (S != null)
{
    byte lineLength;
    ushort lineAddress;
    string lineCode;
    byte[] lineData;
    int 185hecksum=0;
//Parseing the line
//1st length of data
    lineLength = System.Byte.Parse(S.Substring(1, 2),
             System.Globalization.NumberStyles.AllowHexSpecifier);
//2nd 16 LSB or MSB Address
    lineAddress = System.Uint16.Parse(S.Substring(3, 4),
             System.Globalization.NumberStyles.AllowHexSpecifier);
//3rd code of line ("00" data line; "01" end line; "04" 16 MSB address)
    lineCode = S.Substring(7, 2);
//4th Data
    lineData = new byte[(S.Length – 11) / 2];
    if (S.Length – 11 > 0) //If Exists
    {
        for (byte indice = 0; indice < (S.Length – 11) / 2; indice++)
        {
```

```
            lineData[indice] = System.Byte.Parse(S.Substring(9 +
                    (indice * 2), 2), AllowHexSpecifier);
        }
    }
//Checking code
    switch (lineCode)
    {
        case "00": //Programm normal line
            Byte[] forceRestart = { 0xf, 0xf, 0x4, 0xD };
            serialPort1.Write(forceRestart, 0, 4);
            byte[] serialBuffer = { 0xF, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4 };
            serialBuffer[3] = System.Byte.Parse(S.Substring(5, 2),
                    System.Globalization.NumberStyles.AllowHexSpecifier);
            serialBuffer[4] = System.Byte.Parse(S.Substring(3, 2),
                    System.Globalization.NumberStyles.AllowHexSpecifier);
            serialBuffer[5] = (Byte)lineAddressU;
//Data Charge
            for (short indice = 0; indice < lineData.Length; indice++)
            {
                serialBuffer[indice + 6] = lineData[indice];
            }
            short limite = 22;
            if (lineLength <= 8)
            {
                serialBuffer[2] = 0x01;
                limite = 22 - 8;
            }
            checkSum = 0;
            for (short indice = 1; indice < 22; indice++)
            {
                checkSum += serialBuffer[indice];
            }
            serialBuffer[22] = (Byte)(256 - (Byte)checkSum);
            byte[] DLE = { 0x5 };
            serialPort1.Write(serialBuffer, 0, 1);
            for (short indice = 1; indice < limite; indice++)
            {
                if ((serialBuffer[indice] == 4) || (serialBuffer[indice]
                    == 5) || (serialBuffer[indice] == 0xF))
                        serialPort1.Write(DLE, 0, 1);
                serialPort1.Write(serialBuffer, indice, 1);
            }
            serialPort1.DiscardInBuffer();
            if (serialBuffer[22] == 0x4)
                serialPort1.Write(DLE, 0, 1);
            serialPort1.Write(serialBuffer, 22, 1);
            serialPort1.Write(serialBuffer, 23, 1);  //[EOT]
            System.Threading.Thread.Sleep(100);
            Byte[] ACKBuffer = { 0x0, 0x0, 0x0 };
            do
            {
                serialPort1.ReadTimeout = 1000;
                ACKBuffer[0] = (Byte)serialPort1.ReadByte();
                ACKBuffer[1] = (Byte)serialPort1.ReadByte();
                ACKBuffer[2] = (Byte)serialPort1.ReadByte();
                if ((ACKBuffer[0] != (Byte)'A') || (ACKBuffer[1] !=
                    (Byte)'C') || (ACKBuffer[2] != (Byte)'K'))
                {
                    Byte[] forceStop = { 0x4, 0x4 };
                    serialPort1.Write(forceStop, 0, 2);
                }
```

```
            } while ((ACKBuffer[0] != (Byte)'A') || (ACKBuffer[1] !=
                    (Byte)'C') || (ACKBuffer[2] != (Byte)'K'));
                break;
            case "01":  //End Of the File
                break;
            case "04":  //Change of the ADDRU offset
                lineAddressU = lineAddress;
                break;
            default:
                break;
        }
        S = SR.ReadLine();
    }
```

4.2.1.5 Cliente

Al igual que el servidor del laboratorio el cliente del laboratorio remoto WebLab-Robot ha sido desarrollado utilizando los recursos provistos por el RLMS WebLab-Deusto para este cometido.

El cliente del laboratorio remoto WebLab-Robot se ha desarrollado mediante Google Web Toolkit (GWT). Esta tecnología interpreta código de Java y genera código JavaScript. Debido a que el linkador elimina cualquier función que no es utilizada es necesario desarrollar un cliente específico para cada laboratorio remoto, no pudiendo utilizar una implementación genérica. Todos los laboratorios remotos deben registrarse en la lista global del servidor principal de la instancia del RLMS sobre la que se despliegan.

Para el consumo del laboratorio remoto WebLab-Robot se han desarrollado tres clientes que permiten utilizar el laboratorio remoto desde tres perfiles diferentes:

- *Robot-Estándar*: es el perfil genérico de acceso al laboratorio. El estudiante debe proporcionar el firmware que gobierna el comportamiento del robot y puede monitorizar el funcionamiento del mismo mediante la cámara que enfoca la pista del laboratorio (Figura 4.19).

- *Robot-Movement*: mediante un joystick virtual el alumno puede posicionar el robot móvil donde desee (Figura 4.15).

- *Robot-ProgList*: en este perfil se proporcionan una serie de firmwares programados que implementan diferentes comportamientos: seguir la línea, desplazarse aleatoriamente por la pista sin chocarse y girar alternativamente a izquierda y derecha (Figura 4.19).

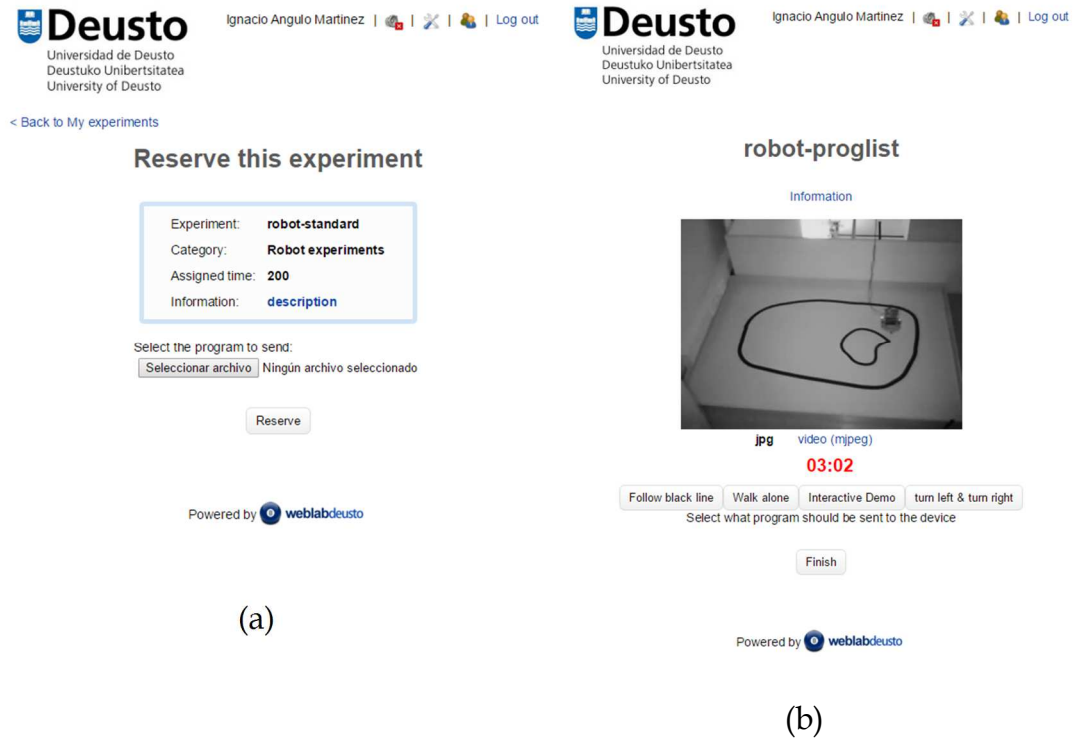

Figura 4.19. Interfaces gráficos de los perfiles "robot-standar" (a) y "robot-proglist" (b) del laboratoprio remoto WebLab-Robot.

4.2.1.6 Servidor de administración

Todas las tareas de administración son llevadas a cabo mediante el sistema de gestión de laboratorios remotos WebLab-Deusto. Este sistema implementa las siguientes funciones de administración (Orduña, y otros, 2011):

- Autenticación/gestión de usuarios.
- Administración de permisos.
- Gestión de colas.
- Registro de trazabilidad de las sesiones de experimentación.
- Balanceo de carga entre diferentes instancias de experimentación.

4.2.2 Experiencia de usuario del laboratorio remoto WebLab-Robot

El éxito de este laboratorio es debido a su sencillez de uso. El interfaz gráfico de usuario proporciona, mediante un wiki, toda la información necesaria para el

desarrollo de experimentos orientados hacia el control del robot móvil sobre el que se fundamenta el laboratorio remoto.

El estudiante puede utilizar cualquier herramienta software para el desarrollo de sistemas basados en microcontroladores PIC, pero debe tener en cuenta el reposicionamiento de los vectores de reset e interrupciones contemplados en el diseño del bootloader (Tabla 4.9). Se proporcionan ficheros de linkaje adaptados para los compiladores de C oficiales de Microchip MPLAB-C18 y MPLAB-XC8, de tal forma que los estudiantes que utilicen estas herramientas pueden omitir las configuraciones especificas requeridas por el diseño del bootloader.

Una vez que el estudiante ha generado el firmware en formato Intel Hex, debe autenticarse en la instancia del RLMS WebLab-Deusto donde se ha desplegado el laboratorio y acceder al laboratorio remoto a través del perfil robot-standard que permite la programación del microcontrolador que gobierna el comportamiento del robot móvil (Figura 4.19). WebLab-Deusto proporciona balanceo de carga entre instancias de experimentación de modo que si todas las disponibles se encuentran ocupadas en el momento del acceso, el gestor de colas accederá a la primera que se libere llegado el turno del estudiante.

Cuando la programación del robot ha finalizado, el cliente muestra el video capturado por la cámara que permite al estudiante validar su experimento.

El estudiante dispone del perfil de acceso WebLab-Movement que permite posicionar el robot en cualquier parte de la pista para validar el código modularmente.

4.2.3 Resumen del laboratorio remoto WebLab-Robot

El laboratorio remoto WebLab-Robot permite a los estudiantes experimentar con un robot móvil a través de Internet siguiendo los mismos pasos que se requieren en los laboratorios de microcontroladores presenciales.

La estabilidad y fiabilidad de este laboratorio ha motivado que haya sido elegido como uno de los laboratorios remotos que permiten a los usuarios invitados comprobar las posibilidades del sistema de gestión de laboratorios remotos WebLab-Deusto.

La Figura 4.20 muestra cómo todos los componentes identificados en la arquitectura abierta propuesta para el despliegue de laboratorios remotos en sistemas embebidos se han implementado siguiendo los requisitos establecidos, integrados con el sistema de gestión de laboratorios remotos WebLab-Deusto. Así

mismo, la instancia del laboratorio remoto WebLab-Robot desplegada en el MIT - Massachusetts Institute of Technology se ha integrado en su propio sistema de gestión de laboratorios remotos "iLab". Estas implementaciones demuestran la capacidad de integración de la arquitectura con plataformas de gestión de laboratorios remotos.

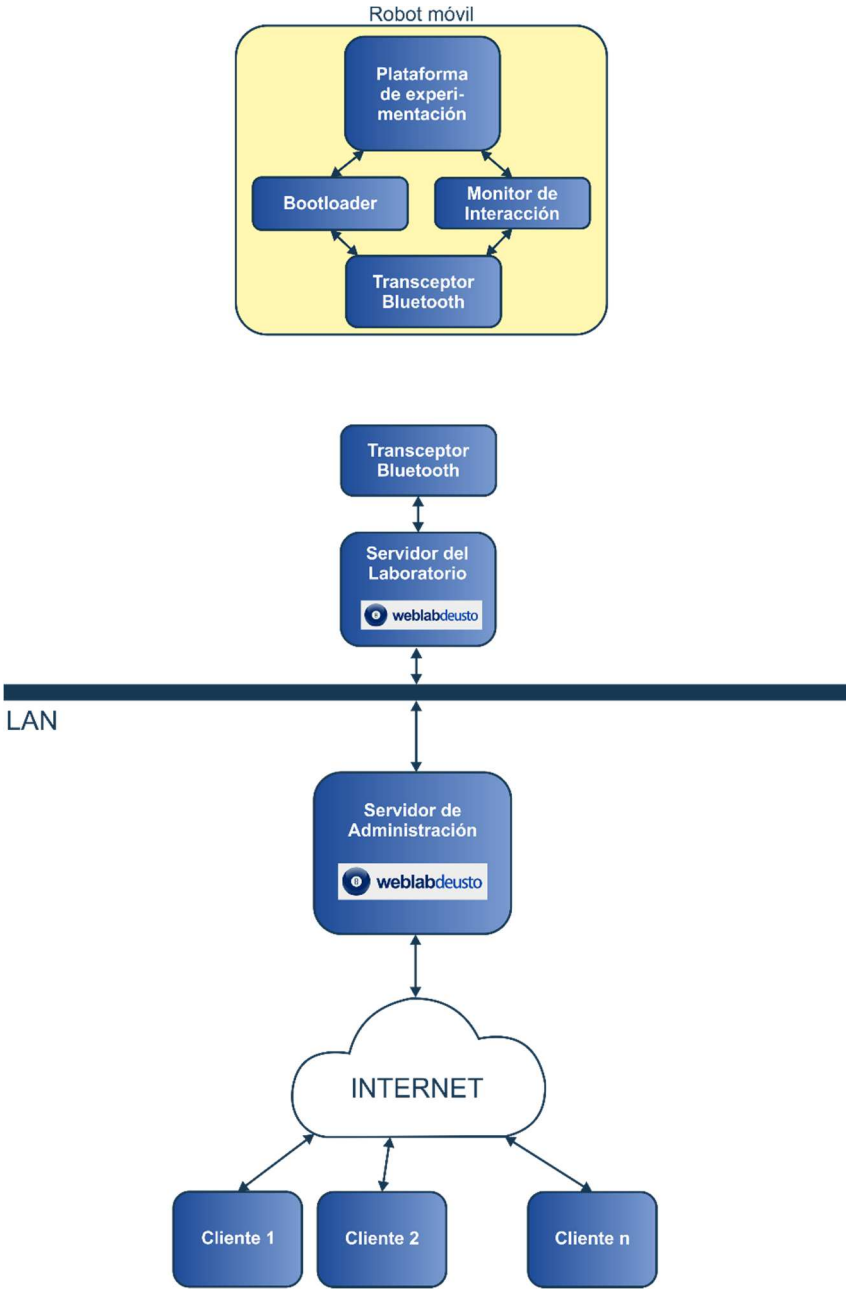

Figura 4.20. Arquitectura del laboratorio remoto WebLab-Robot.

4.3 Plataforma para el despliegue de laboratorios remotos WebLab-Box

El sistema WebLab-Box no es simplemente un laboratorio remoto, es el resultado de integrar todos los componentes identificados en la arquitectura propuesta sobre una plataforma hardware/software profesional que permita el despliegue de cualquier laboratorio remoto para experimentación en sistemas embebidos cumpliendo las características establecidas para ellos y aportando la fiabilidad de un equipamiento industrial (García-Zubia, y otros, 2010) (García-Zubía, Angulo, Dziabenko, & Orduña, 2012).

La plataforma WebLab-Box ha sido diseñada de acuerdo a las especificaciones definidas en la arquitectura abierta para el despliegue de laboratorios remotos con experimentación sobre sistemas embebidos para mejorar la desplegabilidad. Esta plataforma incluye los principales componentes identificados en la arquitectura para el despliegue de un laboratorio remoto diseñados de tal forma que permitan su adecuación a las diferentes tecnologías embebidas. El objetivo es garantizar la desplegabilidad y replicabilidad de los sistemas implementados utilizando la plataforma WebLab-Box y proporcionar solidez y estabilidad a los laboratorios remotos.

Actualmente las instalaciones de WebLab-Deusto disponen de 5 instancias del WebLab-Box que integradas en su RLMS proporcionan experimentación con diferentes tecnologías de sistemas embebidos:

- WebLab-FPGA: Dos instancias idénticas de este laboratorio remoto (WebLab-FPGA1 y WebLab-FPGA2) permiten la experimentación de los estudiantes con la FPGA Spartan-3 de Xilinx (Xilinx, 2009).
- WebLab-CPLD: Dos instancias de este laboratorio permiten la experimentación con un CPLD (complex programmable logic device) XC9574 de Xilinx (Xilinx, 2013).
- WebLab-PIC2: La última instancia de este laboratorio permite la experimentación con el microcontrolador PIC18F45K22 de Microchip (Microchip Technology, 2012).

Mientras que las instancias para experimentación con FPGA y CPLD se han implementado para permitir a los estudiantes de la Universidad de Deusto llevar a cabo la misma experimentación que desempeñan en los laboratorios presenciales, el WebLab-PIC2 implementa un laboratorio a medida del microcontrolador sobre el que se fundamenta para permitir realizar experimentación remota sobre todos los recursos integrados (Figura 4.21).

Adicionalmente una quinta instancia de WebLab-Box (WebLab-MCU) ha sido implementada para su funcionamiento "standalone" sin necesidad de integrarse en ninguna plataforma software. La idea de este laboratorio parte de la construcción de un dispositivo comercializable que no requiera conocimientos técnicos para su despliegue.

Figura 4.21. Cinco instancias de laboratorio remoto remoto para experimentación con diferentes tecnologías de sistemas embebidos basadas en la plataforma WebLab-Box.

Más allá de estos laboratorios remotos desplegados en la Universidad de Deusto, una instancia del laboratorio remoto WebLab-PIC2 ha sido desplegadas en el MIT - Massachusetts Institute of Technology, dentro del proyecto "Building an ecology of Online Laboratories" (Orduña P. , y otros, 2012) subvencionado por la NSF (Award#: 1132813). Además, recientemente una instancia del laboratorio remoto WebLab-FPGA ha sido desplegado en la Universidad Estatal Shota Rustabeli en Batumi, Georgia (Figura 4.22) dentro del proyecto iCo-oP subvencionado dentro del Programa Tempus de la Comunidad Europea (0278-TEMPUS-1-2012-1-DE-Tempus-JPH) (Arras, Henke, Tabunshchyk, & Van Merode, 2015).

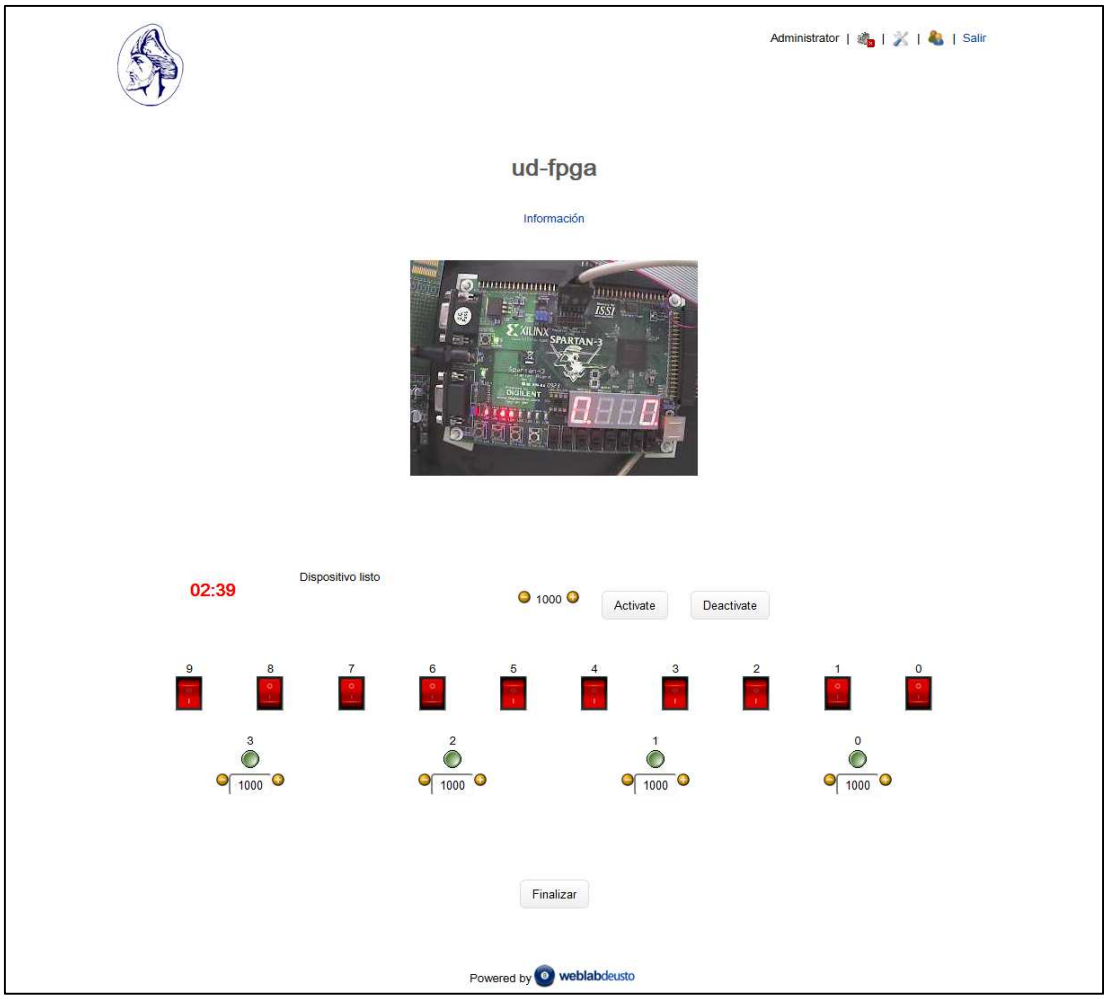

Figura 4.22. Interfaz gráfico de usuario del laboratorio remoto WebLab-FPGA desplegado en la Universidad Estatal Shota Rustabeli en Batumi, Georgia.

Por último, el laboratorio WebLab-MCU desarrollado sobre la plataforma WebLab-Box fue galardonado en el "3rd IEEE Internacional Conference on e-Learning in Industrial Electronics (ICELIE 2009)" con el premio a la mejor herramienta educativa (García-Zubia, y otros, 2009).

4.3.1 Implementación de laboratorios remotos desplegados mediante la plataforma WebLab-Box

La implementación de los laboratorios remotos bajo la plataforma WebLab-Box se fundamenta en la estructura hardware diseñada a medida de la arquitectura definida para albergar los diferentes componentes (Figura 4.23). Esta estructura presenta dos niveles. El nivel superior dispone de espacio para albergar la plataforma de experimentación y el servidor de interacción. Además contiene un sistema de

iluminación propio basado en diodos led y un soporte para la instalación de la Cámara IP que permiten la monitorización óptima de la plataforma de experimentación. En el nivel inferior se albergan el servidor de grabación, el servidor del laboratorio, el sistema que proporciona el cliente y, opcionalmente en aquellos despliegues "standalone", el servidor de administración. Además se incluye una fuente de alimentación multitensión para suministrar energía a todos los sistemas integrados, un sistema propio de ventilación y un switch Ethernet de 5 puertos que permite conectar todos los sistemas a la red local del laboratorio mediante una única conexión Ethernet.

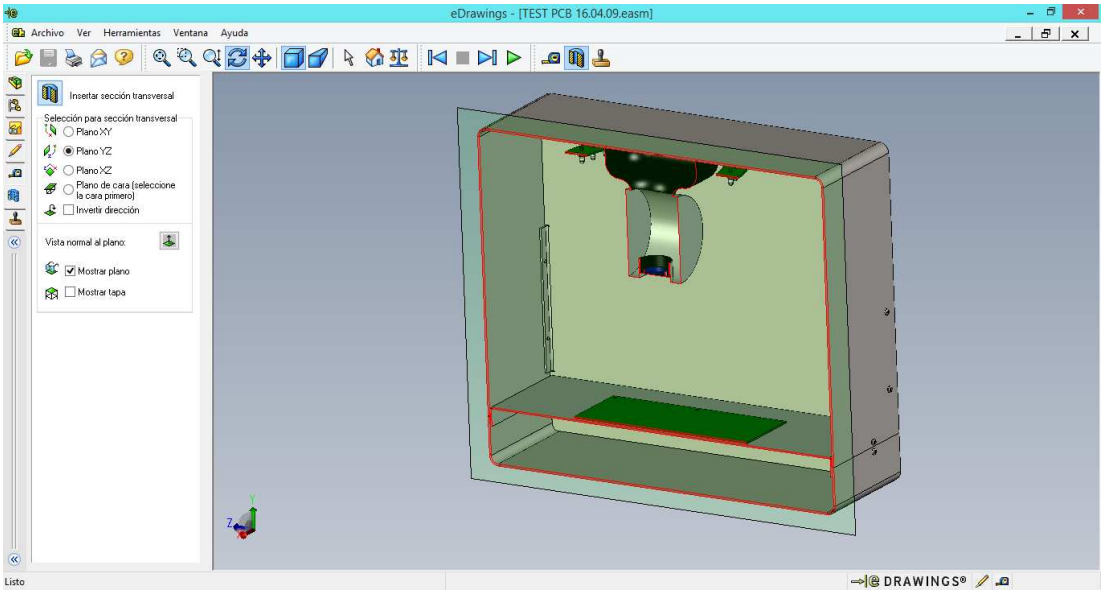

Figura 4.23. Diseño 3D de la estructura hardware WebLab-Box basada en la arquitectura propuesta para el despliegue de laboratorios remotos para experimentación con tecnologías embebidas.

La plataforma WebLab-Box ha sido diseñada a la medida de la arquitectura propuesta. Proporciona implementaciones genéricas y configurables de los componentes comunes a la experimentación con las distintas tecnologías para el desarrollo de sistemas embebidos, y los recursos necesarios para facilitar el despliegue de los componentes específicos que posibilitan una experimentación concreta. La Figura 4.24 muestra la implementación de la arquitectura en la plataforma WebLab-Box. Como se describirán en los siguientes apartados, la plataforma WebLab-Box incluye implementaciones del servidor de administración, el servidor del laboratorio, el cliente y el servidor de interacción. La documentación incluye instrucciones dirigidas al personal encargado del despliegue sobre la configuración de cada componente. Así mismo, se incluye un módulo reconfigurable con un desarrollo genérico para el despliegue del servidor de grabación y la plataforma de experimentación específica para soportar la experimentación deseada.

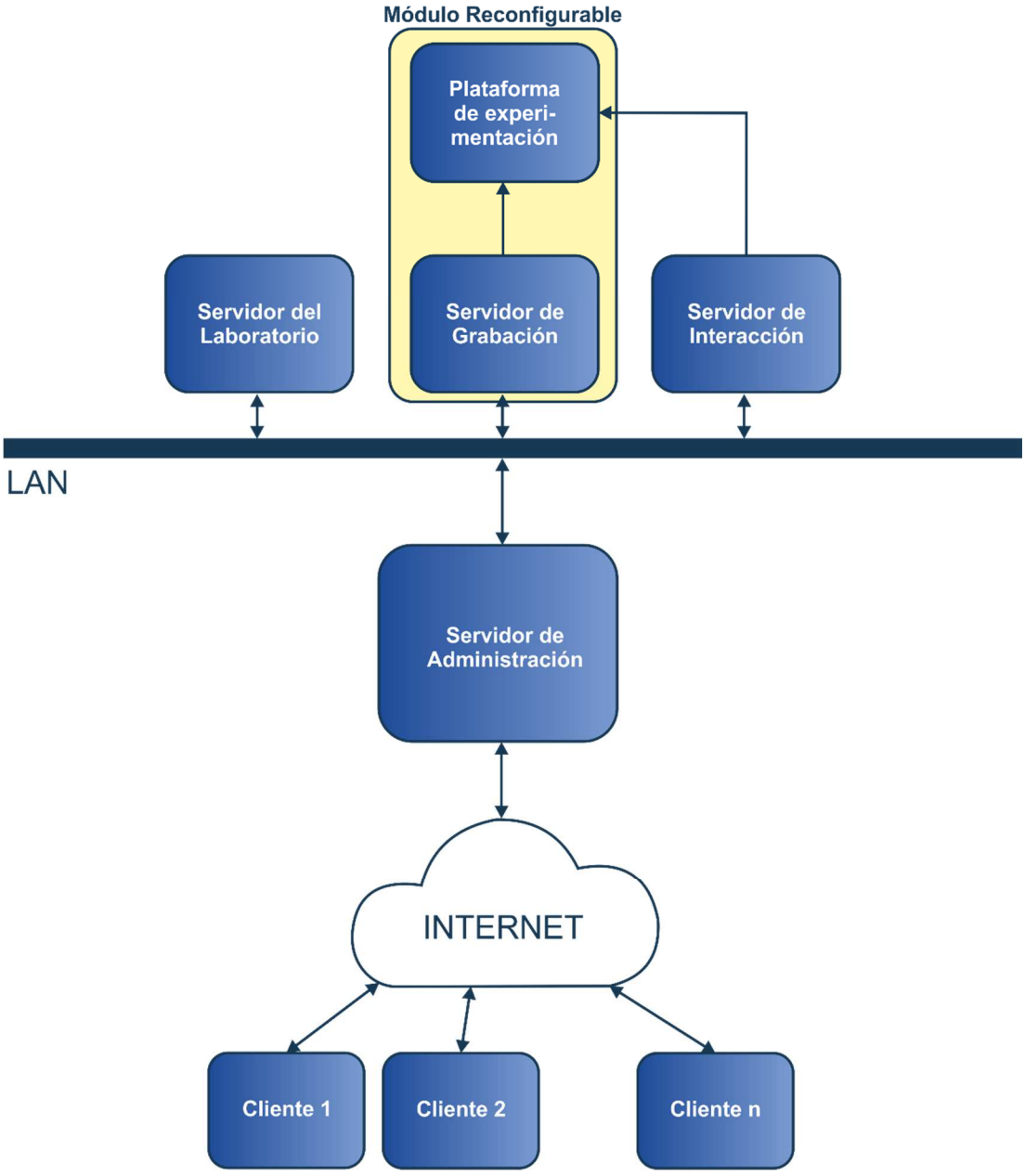

Figura 4.24. Implementación de la arquitectura propuesta en la plataforma WebLab-Box.

4.3.1.1 Servidor de grabación

Una de las ventajas que aporta la plataforma WebLab-Box es su alta adaptabilidad hacia sistemas de grabación. La inclusión de un computador personal multiplataforma de factor de forma reducido que dispone de puertos COM y USB permite adecuar los sistemas de grabación comerciales de las diferentes tecnologías al laboratorio remoto, siguiendo los requisitos establecidos en la arquitectura. Esto simplifica enormemente el desarrollo de uno de los componentes críticos de la

arquitectura, permitiendo el diseño de un servicio Web que recibe el firmware del experimento y lanza la ejecución del software asociado al programador. Además, el switch Ethernet incluido en la plataforma WebLab-Box dispone de una toma adicional para conectar un grabador independiente con conectividad a la red local.

Los programadores utilizados en las instancias desplegadas en las instancias desplegadas en WebLab-Deusto son las siguientes:

- La programación del laboratorio remoto WebLab-FPGA se lleva a cabo mediante un programador Digilent JTAG USB Cable (Digilent, 2010). El software que acompaña a este programador de bajo coste, Digilent Adept (Digilent, 2005) proporciona una API integrable con diferentes tecnologías que facilita el desarrollo del servicio de grabación.

- El servidor de grabación del laboratorio WebLab-PLD está implementado mediante un grabador Ethernet JTAG-Blazer comercializado por Suzaku (Atmark Techno, 2015). El fabricante proporciona un software ejecutable desde línea de comandos que es llamado por el servicio de grabación implementado.

- Las dos instancia para experimentación con microcontroladores, WebLab-PIC2 y WebLab-MCU utilizan el programador PICKit2 de Microchip (Microchip Technology, 2010) para la programación del dispositivo PIC18F45K22. Microchip proporciona una aplicación de código abierto que es utilizada para el desarrollo del servicio de grabación.

La mayoría de los programadores comerciales son integrables en la platafroma WebLab-Box. El servicio Web programado en Python utilizado en las instancias WebLab-FPGA y WebLab-PIC2 permite la ejecución de una aplicación externa para la programación del dispositivo, facilitando el cumplimiento de los requisitos de independencia establecidos en la definición de la arquitectura.

4.3.1.2 Servidor de interacción

La versatilidad que aporta el servidor de interacción desarrollado para el laboratorio remoto WebLab-PIC, descrito en el apartado 4.2.1.2 de este capítulo, lo ha convertido en un elemento fijo de la plataforma WebLab-Box.

El microcontrolador PIC18F97J60 (Microchip Technology, 2011) utilizado para su desarrollo aporta 70 entradas/salidas digitales de propósito general (GPIO), 16 canales ADC, 2 puertos serie síncronos (SPI o I2C), 3 puertos USART, 3 módulos de captura, comparación y modulación de anchura de banda y un puerto paralelo que cumplen con las necesidades de interacción de la mayoría de plataformas de

plataformas de experimentación utilizadas para experimentación remota. Además el sistema de desarrollo utilizado en la WebLab-Box para la implementación del servidor de interacción, PICDEMnet 2 (Microchip Technology, 2006), dispone de una sección de prototipos que permite la integración de elementos adicionales. Por ejemplo, en el caso del laboratorio WebLab-PIC2, que permite la experimentación con el convertidor analógico-digital integrado en el microcontrolador PIC18F45k22 sobre el que se desarrolla la experimentación, esta sección de prototipos sirvió para la integración de un convertidor digital-analógico MCP4902 (Microchip Technology, 2010), que genera la tensión analógica demandada por el estudiante desde el interfaz gráfico de usuario.

4.3.1.3 Plataforma de experimentación

El cumplimiento de la arquitectura propuesta permite la integración de cualquier sistema de desarrollo como plataforma de experimentación de los laboratorios remotos basados en WebLab-Deusto. Las dimensiones de la superficie dispuesta para ubicar la plataforma de experimentación (34cm. x 24cm.) permiten el desarrollo de plataformas a medida basadas en una única tarjeta de circuito impreso o en la integración de varias tarjetas. La cámara robotizada integrada en la plataforma WebLab-Box para la monitorización del experimento permite almacenar las posiciones de los diferentes módulos monitorizables y su enfoque a demanda del servidor del laboratorio (Figura 4.25).

Las plataformas de experimentación utilizadas en los laboratorios remotos desplegados mediante la plataforma WebLab-Box en las instalaciones de laboratorios remotos de la Universidad de Deusto se describen en los siguientes apartados.

4.3.1.3.1 Plataforma de experimentación del laboratorio remoto WebLab-FPGA

Aunque desde su publicación, el laboratorio remoto WebLab-FPGA ha sido utilizado en numerosos cursos dentro y fuera de la Universidad de Deusto (Garcia-Zubia, y otros, 2008), su diseño se llevó a cabo para posibilitar a los estudiantes de la asignatura "lógica programable" cursada en el primer cuatrimestre del tercer curso de Ingeniería Técnica de Automática y Electrónica Industrial. Por este motivo la plataforma de experimentación utilizada es el propio sistema de desarrollo utilizado en el laboratorio presencial de la asignatura, el sistema de desarrollo Spartan 3 Starter Kit fabricado por Digilent (Digilent, 2005).

Figura 4.25. Interior del laboratorio WebLab-CPLD desplegado sobre la plataforma WebLab-BOX.

Al tratarse de un dispositivo diseñado para experimentación presencial, los estudiantes deben sustituir el fichero de restricciones de usuario (UCF) proporcionado por el fabricante por otro desarrollado a medida para el laboratorio remoto WebLab-FPGA. Este fichero sustituye los pines de la FPGA conectados físicamente a los interruptores y pulsadores del sistema de desarrollo por aquellos de los puertos de expansión que se han conectado al servidor de interacción. La utilización de este fichero es la única diferencia que se presenta en la experimentación remota del estudiante respecto a la que desarrolla en el laboratorio presencial.

4.3.1.3.2 Plataforma de experimentación del WebLab-CPLD

El laboratorio remoto WebLab-CPLD es el primer laboratorio remoto publicado por el equipo de investigación WebLab-Deusto en el año 2004 (García Zubía, López de Ipiña, Hernández Jayo, Orduña, & Trueba, 2006). Inicialmente su carácter de prototipo causaba numerosos errores de distinta índole que limitaban la disponibilidad del laboratorio (Figura 4.26). En 2011 este laboratorio fue adaptado a la arquitectura propuesta y desplegado sobre la plataforma WebLab-Box (Garcia-Zubia, y otros, 2011) (Figura 4.25). Durante esta actualización se sustituyó la tecnología embebida sobre la que se fundamenta el laboratorio migrando de un CPLD XC9536 a un CPLD XC9572 con el doble de capacidad pero se respetó la tarjeta de circuito impreso de salidas que permite visualizar las salidas del CPLD mediante diodos led y displays de 7 segmentos. Esto aumenta la inmersión en el laboratorio de los estudiantes de la Universidad de Deusto que cursan asignaturas sobre lenguajes de definición de hardware al monitorizar el resultado de experimentación sobre la misma plataforma utilizada en el laboratorio presencial (Corter, Nickerson, Esche, & Chassapis, 2004).

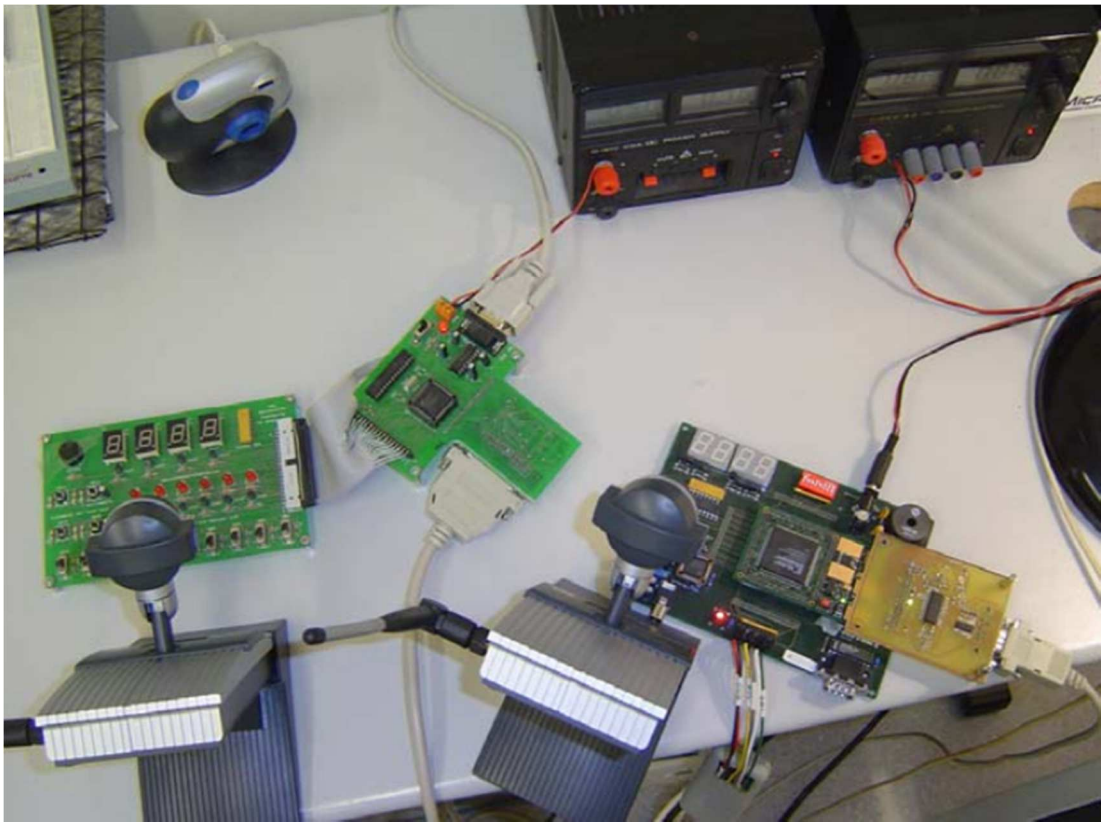

Figura 4.26. Imagen de la primera versión del laboratorio remoto WebLab-PLD.

4.3.1.3.3 Plataforma de experimentación del WebLab-PIC2

A diferencia de en los laboratorios remotos anteriores, en los que la experimentación remota se basa en la desarrollada presencialmente, el diseño de la plataforma de experimentación del laboratorio remoto WebLab-PIC2 se llevó a cabo desde la perspectiva de los requerimientos de un laboratorio remoto. Para mejorar la monitorización a través de la cámara se ha diseñado una tarjeta de circuito impreso de grandes dimensiones que incluye todos los periféricos de salida y un conector compatible con el servidor de interacción para facilitar la conexión de las entradas (Figura 4.27).

Esta plataforma de experimentación está dividida en las siguientes secciones:

- ***Digital Outputs:*** Permite la experimentación con salidas digitales contiene 6 diodos led directamente conectados a 6 GPIOs del microcontrolador
- ***UART Output:*** Dispone de un LCD alfanumérico de 2x16 caracteres con interfaz serie conectado directamente al pin Tx del módulo USART del microcontrolador. Permite visualizar los caracteres ASCII enviados por el puerto serie.
- ***PWM Outputs:*** Dos diodos led de alta luminosidad permiten experimentar con los 2 puertos que permiten la modulación de ancho de banda del microcontrolador.
- ***I2C Com:*** Contiene un expansor de puertos MCP23008 (Microchip Technology, 2007) y un reloj en tiempo real MCP7940 (Microchip Technology, 2011) que permiten al estudiante experimentar con el bus I2C.
- ***SPI Com:*** Dispone de una memoria EEPROM 24LC256 (Microchip Technology, 2013) y un convertidor digital analógico que permiten la experimentación del estudiante con el BUS SPI.
- ***Digital & Analog Inputs:*** Dispone de un conector compatible con el servidor de interacciones para introducir en el microcontrolador las señales demandadas durante la interacción del estudiante.

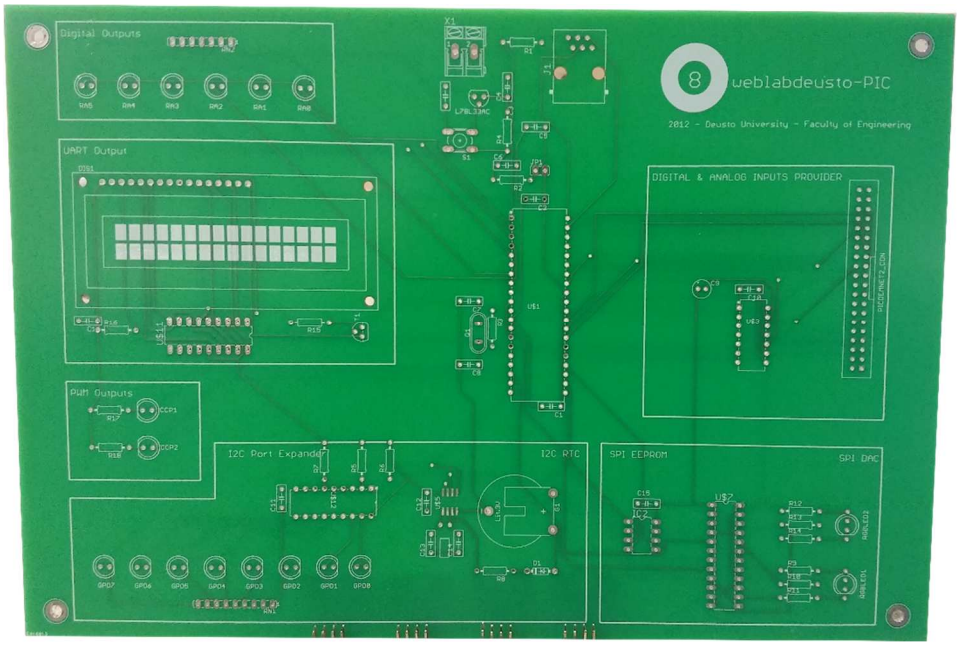

Figura 4.27. Tarjeta de circuito impreso de la plataforma de experimentación del laboratorio remoto WebLab-PIC2.

4.3.1.3.4 Plataforma de experimentación del laboratorio remoto WebLab-MCU

La plataforma de experimentación del laboratorio remoto WebLab-MCU está compuesta por el sistema de desarrollo comercial USB-PIC'School de Microsystems Engineering (Ingeniería de Microsistemas, 2010). Este sistema de desarrollo se utiliza en la asignatura de "Microcontroladores" que se imparte en el segundo parcial del segundo curso de Ingeniería Técnica en Automática y Electrónica Industrial en la Universidad de Deusto.

4.3.1.4 Servidor del laboratorio

La integración de los laboratorios en el sistema de gestión de laboratorios remotos WebLab-Deusto facilita el desarrollo del servidor del laboratorio. El apartado 4.2.1.4 de este capítulo describe el desarrollo de este componente aprovechando los recursos provistos por el RLMS.

Para la instancia del laboratorio remoto WebLab-MCU implementada independientemente de manera "standalone," sin integrarse con ningún sistema de gestión del aprendizaje o de gestión de laboratorios remotos, se ha desarrollado un servidor de laboratorio mediante ASP.NET MVC 2.0 que recibe las órdenes del cliente, desarrollado mediante la misma tecnología, y ejecuta los comandos pertinentes sobre el servidor de grabación o el servidor de interacción.

4.3.1.5 Cliente

Todos los clientes de los laboratorios remotos basados en la plataforma WebLab-Box e integrados en el RLMS WebLab-Deusto han sido desarrollados mediante Google Web Toolkit (GWT) tal y como se describe en el apartado 4.2.1.5 de este capítulo.

Sin embargo, el cliente del laboratorio remoto WebLab-MCU que funciona de manera autónoma se ha desarrollado utilizando ASP.NET MVC2. En este caso tanto el desarrollo web que proporciona el interfaz gráfico de usuario como el servidor de administración se han publicado sobre un servidor Internet Information Server (IIS) de Microsoft Figura 4.28.

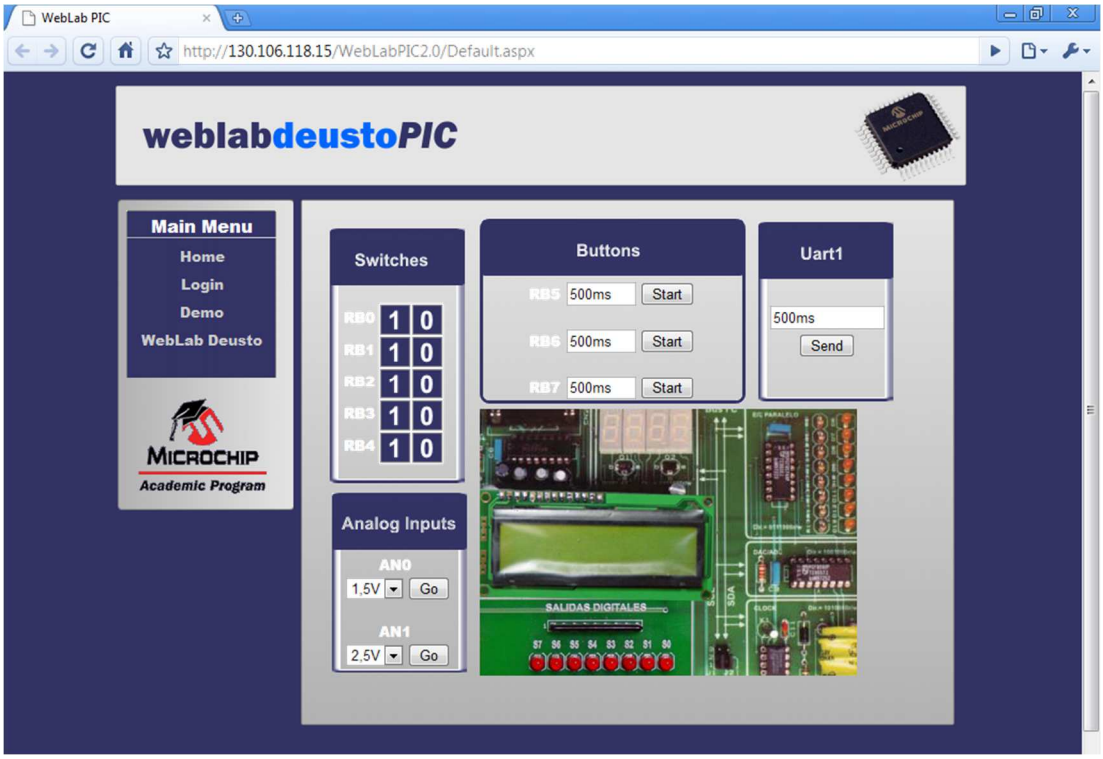

Figura 4.28. Interfaz gráfico de usuario del laboratorio remoto WebLab-MCU.

4.3.1.6 Servidor de administración

El sistema de gestión de laboratorios remotos WebLab-Deusto proporciona todas las características necesarias para la administración de laboratorios remotos. Los servicios de administración son idénticos en todos los laboratorios remotos basados en la plataforma WebLab-Deusto que se encuentran integrados en portal WebLab-Deusto. La Figura 4.29 muestra los diferentes perfiles que permiten acceder a los laboratorios remotos para experimentación sobre tecnologías embebidas desplegados en el portal WebLab-Deusto. Puede observarse como un el RLMS WebLab-Deusto

permite acceder a un mismo laboratorio con diferentes perfiles. Por ejemplo los estudiantes pueden acceder distintamente a las instancias de experimentación del laboratorio remoto WebLab-PLD, ud-pld1 o ud-pld2. Sin embargo, dado que son instancias idénticas también puede acceder a la instancia genérica ud-pld que iniciará una sesión en la primera instancia de WebLab-PLD que quede libre.

Figura 4.29. Portal WebLab-Deusto con los diferentes perfiles que permiten el acceso a los laboratorios remotos desplegados sobre la plataforma WebLab-Box.

La naturaleza de la experimentación en sistemas embebidos, que requiere un tiempo reducido para la experimentación lleva a utilizar un sistema de colas para la gestión del acceso a los laboratorios.

El servidor de administración diseñado a medida para el laboratorio remoto WebLab-MCU ha sido diseñado utilizando ASP.NET MVC2.0. Un sencillo sistema de gestión de colas almacena las peticiones de acceso para su gestión sucesiva. El estudiante no tiene que autenticarse hasta que el laboratorio remoto se libera y se encuentra primera posición de la cola. La ASP.NET Web API2 proporciona servicios básicos de autenticación que son utilizados para el desarrollo de este servidor de administración junto con el framework AngularJS y el sistema de gestión de base de

datos MySQL. Una vez el usuario se autentica accede al interfaz gráfico que puede observarse en la Figura 4.28.

4.3.2 Experiencia de usuario de los laboratorios remotos bajo la plataforma WebLab-Box

Cuatro laboratorios remotos han sido implementados bajo esta plataforma soportando distintas tecnologías embebidas, múltiples instancias paralelas e integración con sistemas de gestión en laboratorios remotos o funcionamiento autónomo.

Todos ellos proporcionan al estudiante los recursos necesarios para desarrollar las etapas de compilación/síntesis del archivo fuente y monitorización e interacción sobre el experimento.

Una peculiaridad interesante del servidor del laboratorio del sistema WebLab-PLD es la capacidad de sintetizar el fichero fuente en el lenguaje de definición de hardware VHDL. El servicio que recibe el fichero enviado por el estudiante, discrimina si se trata de un fichero binario (.bit) ya sintetizado para el dispositivo CPLD que puede ser grabado directamente sobre el CPLD o si por el contrario se trata de un fichero de texto con el código VHDL del experimento del estudiante. En este último caso, para los ficheros con extensión .vhd, el servidor del laboratorio del WebLab-PLD lanza la herramienta de sintetizado Xilinx PlanAhead utilizando el fichero de restricciones de usuario (.ucf) diseñado para el laboratorio remoto y si el proceso resulta exitoso lleva a cabo la programación en el CPLD a través de la herramienta Xilinx Impact. En caso contrario reporta al estudiante el fichero de error generado durante el proceso fallido de síntesis. Esta capacidad aporta un alto índice de universalidad al laboratorio ya que el estudiante únicamente requiere un editor de texto y un navegador de Internet para poder llevar a cabo todas las etapas de la experimentación con sistemas embebidos, incluyendo la edición, la síntesis y el mapeo del experimento. La limitación física de los experimentos desarrollables por un CPLD agiliza el proceso de síntesis que en el caso de la síntesis de una FPGA o de la compilación de un microcontrolador conllevaría un importante aumento en los tiempos de la sesión.

La fiabilidad de esta plataforma se demuestra comprobando la disponibilidad de los laboratorios desplegados bajo su uso. Por ejemplo los laboratorios remotos WebLab-PLD y WebLab-PFGA, llevan funcionando ininterrumpidamente desde el año en que se produjo su despliegue bajo WebLab-Box sin apenas mantenimiento y sufriendo únicamente los parones motivados por los cortes eléctricos de la instalación

o por las mudanzas sufridas por el laboratorio WebLab-Deusto durante estos años. Entre ambos suman más de 30.000 sesiones de experimentación.

4.3.3 Resumen de la plataforma WebLab-Box

La plataforma WebLab-Box ha sido diseñada para proporcionar un equipo profesional para el despliegue de laboratorios embebidos para experimentación sobre tecnologías embebidas siguiendo la arquitectura propuesta en esta tesis.

Su principal objetivo es permitir el despliegue de laboratorios remotos eficaces y fiables que permitan desarrollar experimentación sobre cualquier tecnología embebida por profesionales y educadores en sistemas embebidos que no tengan experiencia en el desarrollo de laboratorios remotos.

WebLab-Box proporciona todos los componentes hardware/software necesarios para remotizar el acceso de cualquier sistema de desarrollo de sistemas embebidos, así como toda la documentación para su despliegue en cualquier instalación.

Soporta integración del laboratorio remoto en sistemas de gestión de laboratorios remotos pero también proporciona una configuración "standalone" que facilita el despliegue para profesionales con conocimientos de informática limitados.

4.4 Laboratorio remoto WebLab-Elevator

WebLab-Elevator es el último de los laboratorios remotos que ha sido desarrollado para la validación de la arquitectura propuesta en esta tesis. Utiliza el misma sistema de desarrollo que el laboratorio remoto WebLab y en la su implementación se han reutilizado la mayoría de sus componentes (Angulo, García Zubia, & Asunción, 2014). Sin embargo, presenta una diferencia que hemos considerado relevante para la validación de la arquitectura abierta de fácil despliegue para permitir la experimentación con sistemas embebidos desde laboratorios remotos: *En lugar de desarrollar experimentación sobre sistemas educativos, permite el control de un equipamiento industrial.*

El empleo de sistemas de desarrollo educacionales y entrenadores didácticos permite llevar a cabo experimentación real con diferentes tecnologías en el ámbito de la microelectrónica y los sistemas embebidos. Sin embargo, el propio carácter didáctico y en ocasiones multidisciplinar de estos equipos conlleva una relajación en las especificaciones de los requisitos de los experimentos que se llevan a cabo. La simulación de los periféricos de entrada y salida mediante dispositivos de carácter

general (diodos led, potenciómetros, motores didácticos, pulsadores e interruptores, etc…) aleja al estudiante de las restricciones temporales exigidas por una instalación real. En este sentido el empleo maquetas industriales proporciona unos requerimientos idénticos a los existentes en ámbitos industriales generando idénticos problemas y errores a los que los ingenieros se enfrentan en su carrera profesional. El elevado coste de los modelos industriales, desarrollados con sistemas profesionales, dificulta la adquisición y mantenimiento de los mismos por los centros educativos así como el acceso regular de los estudiantes. Esta realidad proporciona un escenario idóneo para la utilización de laboratorios remotos que permiten a los alumnos llevar a cabo experimentación real sobre equipamiento didáctico de alto coste, de una forma similar a la que se emplea en laboratorios presenciales (Angulo & García-Zubia, 2014).

4.4.1 Implementación del laboratorio remoto WebLab-Elevator

La maqueta en la que se fundamenta este experimento remoto, desarrollada por la empresa Staudinger GMBH (Figura 4.30), incluye una instalación miniaturizada de un ascensor de tres niveles que incluye todos los equipos de seguridad y funcionamiento esenciales: grupo tractor de dos velocidades, finales de carrera, sensores de aproximación y llegada a cada nivel, sensores de apertura de puertas, sistemas de emergencia, sistema neumático para la apertura de puertas, panel de control en cabina, pulsadores de llamada en cada nivel, indicadores luminosos de nivel, etc.

El control de todos los sistemas integrados en la maqueta se lleva a cabo mediante entradas (26) y salidas (24) digitales a 24V controlables desde sendos conectores SUB-DB37 incluidos en la maqueta.

La naturaleza digital de todos los sensores y actuadores incluidos en la maqueta, permite desarrollar la lógica que gobierna el sistema utilizando diferentes tecnologías: microcontroladores, CPLD, FPGA, PLC o software de control instrumental (LabView, etc.). Sin embargo, dado que el objetivo principal para el que se ha llevado a cabo la implementación de este laboratorio es validar la adecuación de la arquitectura propuesta en esta tesis en el desarrollo de experimentos orientados al control de equipamiento industrial, y para facilitar las tareas de implementación, se utilizará para el control de la maqueta el mismo sistema de desarrollo utilizado en el laboratorio WebLab-FPGA.

El laboratorio remoto permite al alumno desarrollar la lógica del experimento mediante un lenguaje de definición de hardware, sintetizar su diseño para la FPGA SPARTAN -3 (Digilent, 2005) y a través de la plataforma para la gestión de laboratorios remotos WebLab-Deusto programar el dispositivo y verificar el correcto

funcionamiento utilizando un interfaz gráfico que permite monitorizar el ascensor e interactuar con los diferentes sistemas de control incluidos desde un navegador Web.

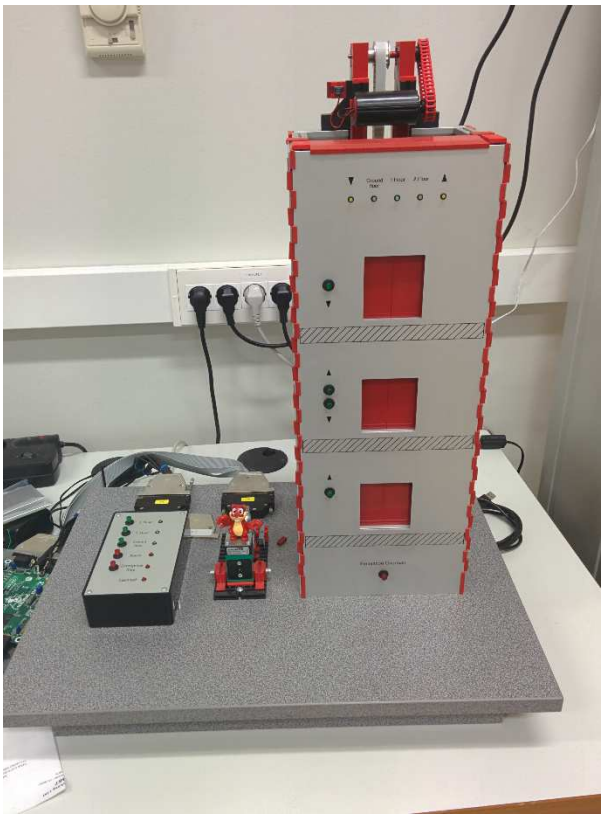

Figura 4.30. Maqueta didáctica Nº 220006 de Staudinger GmbH.

Como se ha comentado anteriormente, debido a que el objetico de este laboratorio remoto es validar la arquitectura en experimentación sobre equipamiento industrial, la implementación de este laboratorio remoto se ha llevado a cabo teniendo como base el sistema WebLab-Elevator, reutilizando parte de los componentes que completan la arquitectura.

Sobre una réplica del sistema WebLab-FPGA se han modificado los componentes que permiten adecuar la maqueta didáctica para su control desde la FPGA. Además de cambiar el cliente para adecuarlo a la nueva experimentación ofrecida por el laboratorio, se han tenido que añadir un conjunto de componentes que permiten trabajar con los niveles industriales exigidos por la maqueta a controlar. Estos componentes que afectan fundamentalmente a la plataforma de experimentación, pueden observarse en color verde sobre la Figura 4.31.

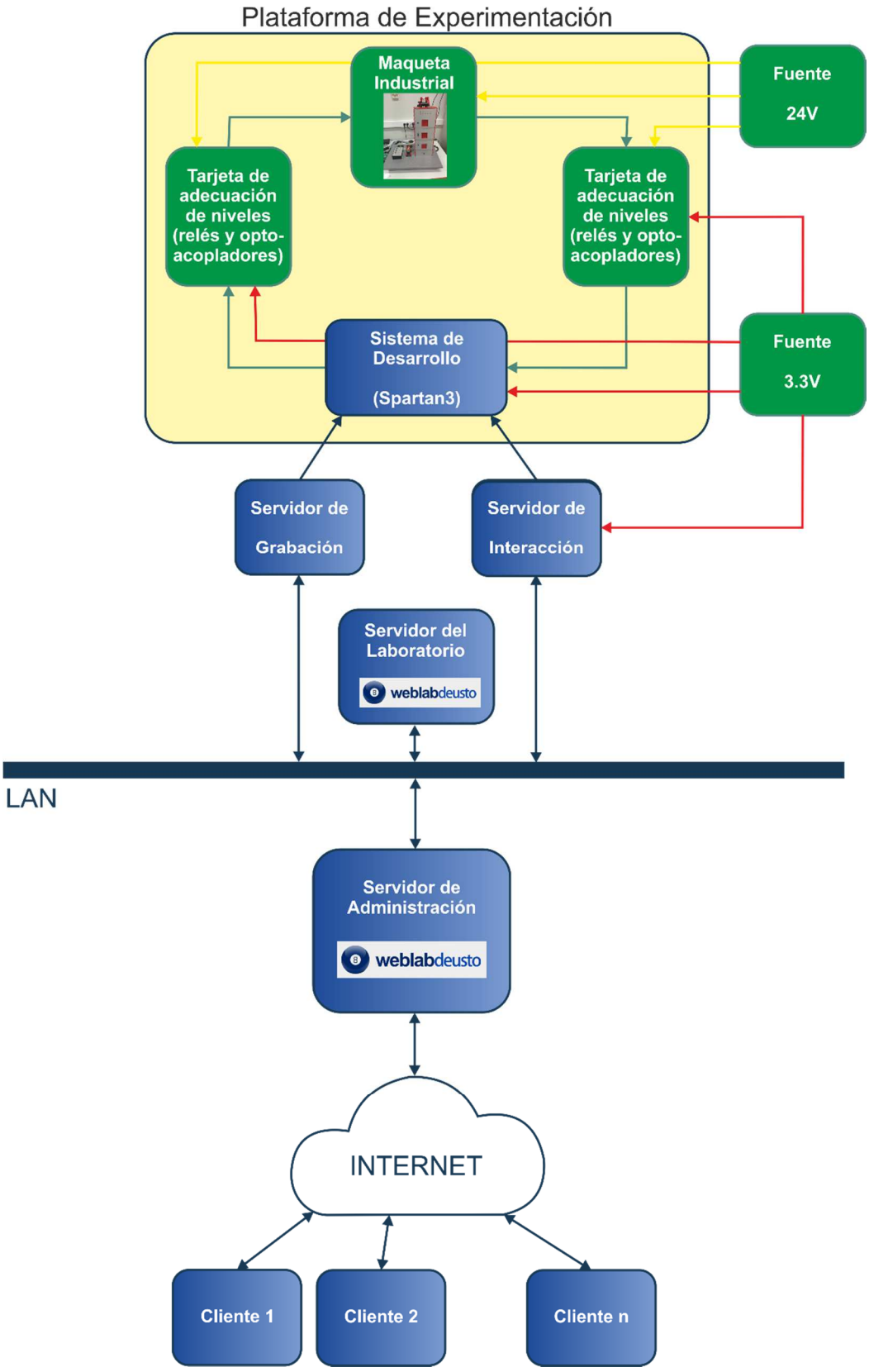

Figura 4.31. Arquitectura del laboratorio remoto WebLab-Elevator.

4.4.1.1 Plataforma de experimentación del laboratorio remoto WebLab-Elevator

La plataforma de experimentación implementada para el laboratorio remoto WebLab-Elevator requiere los siguientes elementos (Figura 4.32):

- Sistema de desarrollo FPGA
- Maqueta industrial de un ascensor de tres plantas
- Tarjetas de adecuación de niveles

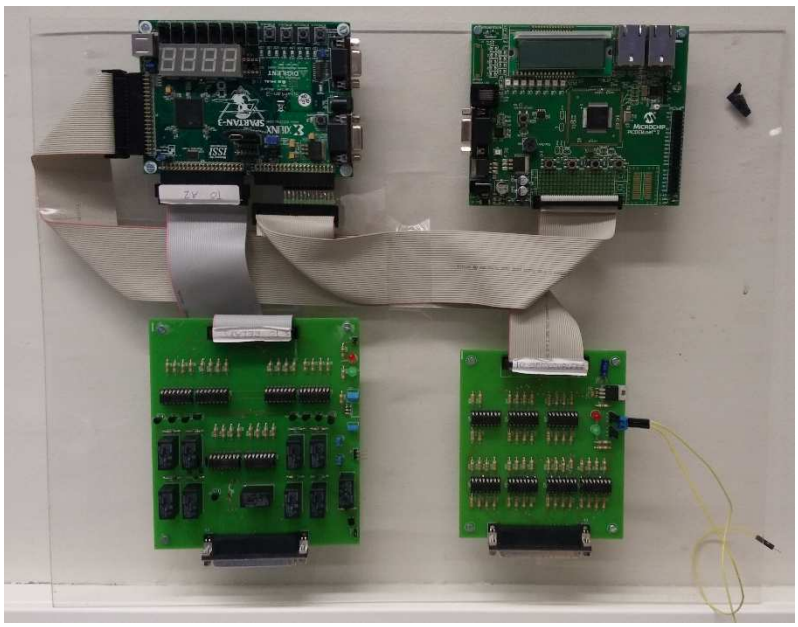

Figura 4.32. Plataforma de experimentación del laboratorio remoto WebLab-Elevator.

4.4.1.1.1 Sistema de desarrollo para FPGA del laboratorio remoto WebLab-Elevator

La plataforma embebida elegida para permitir al estudiante experimentación sobre FPGA es el sistema de desarrollo Spartan 3 Starter Kit (Digilent, 2005), utilizado también en el laboratorio remoto WebLab-FPGA. Este sistema cuenta con tres puertos de expansión de 40 pines cada uno que permiten organizar de forma ordenada las entradas digitales que vienen desde el ascensor (1), las salidas digitales dirigidas al mismo (2) y finalmente las entradas provenientes desde el microservidor que genera las señales demandadas por el estudiante desde el interfaz gráfico (3).

4.4.1.1.2 Maqueta industrial de un ascensor de tres plantas del laboratorio remoto WebLab-Elevator

La maqueta correspondiente al ascensor de tres plantas simula una instalación de ascensor, equivalente a cualquier ascensor instalado en plantas industriales de varios niveles con todos los equipos de seguridad y de funcionamiento esenciales.

El ascensor consta de una cabina de ascensor con accionamiento por correa, un sistema de elevación y tres unidades de planta. Cada planta dispone de un sistema

neumático de deslizamiento de puertas alimentado por un compresor de miniatura, botones de llamada e indicadores luminosos de color para indicar la dirección del movimiento de la cabina. Interruptores mecánicos detectan la llegada de la cabina a cada planta permitiendo gestionar el movimiento lento-rápido de la cabina. Cada puerta dispone de una barrera luminosa de seguridad para la detección de elementos que puedan obstruir el cerrado. Además la maqueta dispone de un panel de control, desde el cual se pueden realizar las acciones de funcionamiento desde el interior de la cabina: pulsadores de selección para elegir un piso, pulsador de alarma, pulsador de emergencia y un conmutador que permite establecer un modo de funcionamiento en el cual el ascensor es controlado exclusivamente desde fuera de la cabina.

4.4.1.1.3 Tarjetas de adecuación de niveles del laboratorio remoto WebLab-Elevator

Para la adecuación de los niveles de tensión soportados por los sensores y actuadores de la maqueta (24V) y los niveles de las entradas y salidas de propósito general (GPIO) del sistema de desarrollo para FPGA (3,3V) se ha desarrollado dos tarjetas de circuito impreso a medida. Aunque durante el acceso remoto a la maqueta no se utilizarán los 10 pulsadores de control incluidos en la misma, se ha procedido a la inclusión de los mismos para permitir también el acceso presencial al experimento y facilitar las tareas de mantenimiento.

- *Tarjeta de adecuación de sensores*: El ascensor genera 26 salidas digitales que se corresponden con los sensores lumínicos de cada puerta, los interruptores mecánicos y los pulsadores de control. Estas salidas han sido conectadas a la tarjeta de desarrollo mediante el empleo opto-acopladores que permiten asilar eléctricamente ambos dispositivos. El circuito integrado PS-2501-4 integra cuatro opto-acopladores con salida a transistores conmutables con una corriente de 50mA.
- *Tarjeta de adecuación de actuadores*. El funcionamiento del ascensor se gestiona mediante 24 salidas digitales. 12 de ellas se corresponden con indicadores luminosos que pueden ser gobernados directamente desde opto-acopladores de forma análoga a la llevada a cabo en la tarjeta de adecuación de sensores. Además el ascensor dispone de un relé que activa el compresor, 6 válvulas que permiten la apertura y cerrado de las puertas correderas de cada planta y 4 salidas que permiten la gestión del motor que desplaza el ascensor. Estas 12 últimas líneas son gobernadas mediante relés que proporcionan 24Vcc directamente desde la fuente de alimentación.

4.4.1.2 Cliente del laboratorio remoto WebLab-Deusto

El desarrollo del cliente del laboratorio remoto se ha llevado a cabo utilizando el nuevo soporte de javascript incluido en el sistema de gestión de laboratorios remotos

WebLab-Deusto, que permite el desarrollo de clientes gestionados ("managed") utilizando esta tecnología y aporta mucha más libertad que Google Web Toolkit a la hora de disponer los elementos en el cliente.

El cliente dispone de tres cámaras que muestran la parte trasera del ascensor, la parte frontal y la botonera de cabina. Además el estudiante puede accionar todos los pulsadores dispuestos en la maqueta para el control del ascensor desde botoneras virtuales (Figura 4.33).

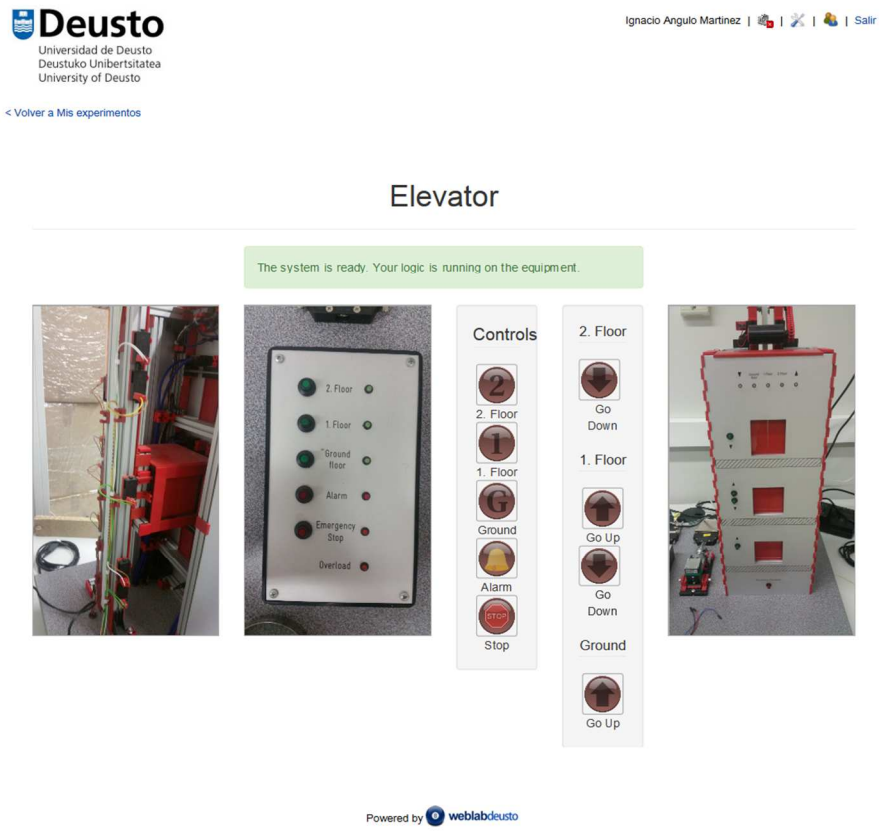

Figura 4.33. Cliente del laboratorio remoto WebLab-Elevator.

4.4.2 Escenario de uso del laboratorio remoto WebLab-Deusto

La experimentación desarrollada por el estudiante sobre el experimento WebLab-Elevator se desarrolla siguiendo el mismo proceso requerido en un laboratorio presencial. El estudiante necesita disponer de un computador con el software de diseño WebPack ISE, suministrado gratuitamente por Xilinx. Esta solución software incluye un conjunto de herramientas que permiten editar el código VHDL, sintetizar

el sistema para la FPGA Spartan-3 y simular el funcionamiento del sistema mediante el completo Xilinx Simulator.

El estudiante puede descargar el fichero de restricciones de usuario (UCF) que contiene la dirección física en la FPGA de cada entrada y salida del sistema y desarrollar por completo la lógica que gestiona el funcionamiento del ascensor o descargar también un fichero de ejemplo con el funcionamiento básico del ascensor y completar la funcionalidad del mismo.

Una vez el estudiante completa el diseño del sistema y genera el fichero de programación desde la herramienta WebPack ISE, entonces debe acceder a la plataforma WebLab-Deusto, autenticarse y seleccionar el experimento WebLab-Elevator.

Nada más comienza la sesión de experimentación el estudiante debe seleccionar el fichero binario generado y WebLab-Elevator lleva acabo la programación del dispositivo. Cuando este proceso concluye, el estudiante puede verificar el funcionamiento de su diseño desde el interfaz de usuario mostrado en la Figura 4.33.

4.4.3 Resumen del laboratorio remoto WebLab-Elevator

El empleo de laboratorios remotos basados en maquetas industriales permite llevar a cabo prácticas de alto nivel con grupos numerosos de alumnos. Las características en el aprendizaje de sistemas embebidos (Grimheden & Törngren, 2005), donde el alumno puede desarrollar en su ordenador personal el firmware a desplegar en el sistema, junto con el tiempo reducido que conlleva la validación del sistema desplegado sobre la propia maqueta, facilita la compartición del experimento mediante un laboratorio remoto.

La arquitectura abierta para el despliegue de laboratorios remotos con experimentación sobre sistemas embebidos permite la publicación en internet de experimentos basados tecnologías embebidas para el control de dispositivos industriales. Debido al alto coste de estos equipamientos y la dificultad para organizar la experimentación con ellos en los laboratorios presenciales, los laboratorios remotos se postulan como una opción óptima que permite la experimentación segura, controlando el tiempo asignado a cada estudiante y permitiendo el acceso a dominios importantes de estudiantes.

Esta implementación se ha llevado a cabo para evaluar la adaptabilidad de la arquitectura ante los requisitos impuestos en el control de equipamiento industrial. Todos los sistemas de alimentación y adecuación de niveles requeridos para el control

de la maqueta didáctica de forma segura han podido ser incluidos en la arquitectura cumpliendo las especificaciones de la misma.

4.5 Resumen de las implementaciones

A lo largo del capítulo 4 se han descrito 4 implementaciones llevadas a cabo para la validación de la arquitectura desde diferentes perspectivas.

La primera implementación, el laboratorio remoto WebLab-PIC, permitió demostrar la posibilidad de implementar los distintos componentes identificados en la arquitectura, siguiendo las especificaciones descritas en el capítulo 3, sobre dispositivos embebidos de bajo rendimiento. Esta prueba de concepto demuestra la posibilidad de implementar laboratorios remotos de bajo coste para la experimentación con sistemas embebidos. Todos los componentes del laboratorio fueron integrados en dos microcontroladores de 8 bits, implementando un sistema autónomo capaz de ofrecer experimentación con microcontroladores a través de Internet con un coste aproximado de 200€. El desarrollo de cada uno de los componentes se llevó a cabo persiguiendo la funcionalidad y conectividad establecidas en la descripción de la arquitectura. Cabe destacar el desarrollo de un servidor de grabación basado en un bootloader a través de protocolo TFTP que permite la auto-programación del dispositivo sobre un firmware genérico sin configuraciones específicas debidas al bootloader.

La implementación del segundo laboratorio remoto, WebLab-Robot, se llevó a cabo para validar la integrabilidad de los laboratorios basados en la arquitectura propuesta en plataformas de experimentación o aprendizaje. El diseño de los componentes que conforman la arquitectura se realizó utilizando los recursos provistos por el sistema de gestión de laboratorios remotos WebLab-Deusto validando la capacidad de implementar laboratorios remotos basados en la arquitectura propuesta completamente integrables en un RLMS.

El objetivo de la tercera implementación fue diseñar un sistema que facilitara el despliegue de laboratorios remotos para experimentación con cualquier tecnología de desarrollo de sistemas embebidos, aportando la fiabilidad de un sistema profesional. La plataforma WebLab-Box incluye los principales componentes identificados en la arquitectura y permite adaptar distintas plataformas de experimentación, proporcionando los recursos necesarios para la administración del laboratorio remoto. La validación de la plataforma se llevó a cabo mediante la implementación de tres laboratorios remotos, WebLab-CPLD, WeLab-FPGA y WebLab-PIC2, que permiten la experimentación remota sobre tres tecnologías diferentes utilizando la plataforma WebLab-Box.

Finalmente la última implementación se llevó a cabo para demostrar la capacidad de los laboratorios remotos implementados basados en la arquitectura propuesta de controlar dispositivos de naturaleza industrial en lugar de didáctica. El laboratorio remoto WebLab-Elevator permite el control de una maqueta didáctica basada en un ascensor de tres plantas desde una FPGA, validando la adecuación de la arquitectura a los diferentes dispositivos que son controlables desde sistemas embebidos.

Capítulo

5

Validación de la arquitectura propuesta

Durante el estado del arte llevado a cabo en el capítulo 2, y fruto del análisis de los principales sistemas desplegados, se han identificado las principales características que deben cumplir los laboratorios remotos que ofrecen experimentación con tecnologías relativas a sistemas embebidos. En el capítulo 3 se ha presentado una arquitectura que identifica los componentes requeridos para el despliegue de un laboratorio remoto y la forma en la que deben ser conectados para facilitar el despliegue y la experimentación remota sobre esta tecnología. Posteriormente en el capítulo 4 se han descrito un total de 7 laboratorios remotos implementados siguiendo la arquitectura propuesta que permiten la experimentación con diversas tecnologías de sistemas embebidos. Finalmente a lo largo del presente capítulo se validará la arquitectura propuesta mediante el análisis de las características identificadas como fundamentales para los laboratorios remotos que centran este trabajo en las implementaciones descritas.

Es importante recalcar que el propósito de la arquitectura planteada no se extiende a la experimentación remota con carácter universal, enfocándose únicamente en aquellos que permiten la experimentación sobre las distintas tecnologías orientadas al desarrollo de sistemas embebidos. En el capítulo 1 se ha

recalcado el auge que estos sistemas están experimentando en el los últimos años y la importancia de los laboratorios remotos para el aprendizaje en el desarrollo de sistemas embebidos. En este sentido 3 características particulares han sido identificadas como fundamentales para los laboratorios remotos que permiten la experimentación con estas tecnologías:

- *Replicabilidad*: El rápido avance en estas tecnologías requiere el continuo reemplazo de los dispositivos con los que se experimenta siendo fundamental que los laboratorios remotos faciliten su adecuación a los nuevos modelos y familias de dispositivos que aparecen.
- *Escalabilidad*: La naturaleza de los dispositivos embebidos exige su programación para el desarrollo de experimentación real. El tiempo necesario para realizar esta programación, añadido al tiempo de interacción requerido para la validación del sistema complica el acceso a los laboratorios por grupos elevados de estudiantes. La solución a este problema es disponer de múltiples instancias de experimentación que permitan el acceso concurrente de varios estudiantes.
- *Desplegabilidad*: La demanda de laboratorios remotos que permitan la experimentación con sistemas embebidos contrasta con la complejidad de los conocimientos requeridos para su desarrollo. Por ello es importante que la comunidad investigadora dedicada al desarrollo de laboratorios remotos contemple el facilitar su posterior despliegue en otras instalaciones por personal con un único perfil de consumidor. Durante el estado del arte se han analizado 25 laboratorios remotos que han sido desarrollados por las propias universidades que se benefician de su consumo. Aunque muchos de ellos permiten el acceso a estudiantes de otras instituciones, el verdadero camino para la democratización de la experimentación remota consiste en permitir que las propias instituciones que ofrecen docencia en sistemas embebidos puedan desplegar y administrar laboratorios remotos que complementen la educación ofrecida.

Aunque el diseño de la arquitectura se ha conducido fundamentalmente a la consecución estas 3 características, para la validación es importante no solo demostrar su cumplimiento, sino la manera en la que el cumplimiento de la arquitectura afecta en el resto de características que determinan el funcionamiento de los laboratorios remotos. En este sentido también se analizarán las siguientes características en los laboratorios implementados:

- *Capacidad para integración con plataformas software que facilitan el despliegue de laboratorios remotos*. Los sistemas de gestión de laboratorios remotos ofrecen recursos destinados a mejorar la experiencia de usuario y la seguridad de los laboratorios remotos que se integran en estas plataformas software. Además

como se ha demostrado (San Cristóbal, 2010) los sistemas de gestión del aprendizaje facilitan el acceso de los estudiantes a los laboratorios remotos. Además la integración con este tipo de plataformas software permite la correcta compartición de los laboratorios remotos entre diferentes instituciones (Orduña, 2013). Además las comunidades de desarrolladores que se encargan de la evolución de estas plataformas fomentan la sostenibilidad de los laboratorios.

- *Disponibilidad*. Una de las principales ventajas de los laboratorios remotos es que permiten a los estudiantes llevar la experimentación en cualquier momento y desde cualquier lugar. Los laboratorios remotos deben de permitir gestionar las sesiones de experimentación y proporcionar los recursos para el desarrollo de las mismas de forma ordenada, ofreciendo una buena experiencia de usuario.
- *Conectividad*. Además de facilitar el despliegue de los laboratorios remotos es fundamental utilizar tecnologías que no dificulten la conectividad de los sistemas en todo tipo de infraestructuras sin perjudicar la seguridad de las instalaciones.
- *Universalidad*. Los cambios sociales deben ser tenidos en cuenta en el diseño de laboratorios remotos. En España, la Encuesta sobre Equipamiento y Uso de Tecnologías de Información y Comunicación en los Hogares llevada a cabo por el Instituto Nacional de Estadística en el año 2014 (INE, 2014) revela por primera vez que los dispositivos móviles constituyen ya la principal puerta de acceso a la red. La accesibilidad de los laboratorios desde dispositivos móviles es un factor determinante (Orduña P. , García-Zubia, Irurzun, & Rodriguez-Gil, 2011).

5.1 Metodología para la validación de la arquitectura mediante el análisis de los laboratorios remotos implementados

Establecidas las características más relevantes para el rendimiento de los laboratorios remotos implementados para soportar experimentación sobre tecnologías de desarrollo de sistemas embebidos, la validación de la arquitectura propuesta se llevará a cabo evaluando el grado de cumplimiento de dichas características sobre los laboratorios remotos implementados bajo las especificaciones descritas en la descripción de la arquitectura. Dado que el grado de cumplimiento para cada característica sólo puede evaluarse de una forma cualitativa, la validación se llevará a cabo mediante una comparativa de los laboratorios remotos implementados utilizando la arquitectura propuesta, con aquellos analizados durante el estado del arte.

La metodología utilizada para la evaluación del cumplimiento de todas estas características va a realizarse desde una doble vertiente. Inicialmente se analizará el cumplimiento de las características en cada una de las implementaciones descritas en el capítulo 4, analizando los factores particulares de cada laboratorio remoto. Posteriormente se evaluará el grado de desempeño general de cada una de las características en el conjunto de los laboratorios implementados según la arquitectura propuesta atendiendo a factores generales que dependen de la arquitectura.

Para la evaluación de la desplegabilidad de los laboratorios remotos implementados se analizarán los componentes hardware/sofware diseñados a medida que puedan comprometer el despliegue de nuevas instancias desde otras instituciones. En este punto se hará especial hincapié en los despliegues de las implementaciones llevadas a cabo siguiendo la arquitectura planteada que hayan tenido lugar desde terceras instituciones. La existencia de casos de éxito en los que un laboratorio remoto ha podido ser desplegado en otra institución representa en sí mismo un modo empírico que permite la validación de la desplegabilidad de los laboratorios remotos implementados.

La característica de replicabilidad de las tecnologías sobre las que se desarrolla la experimentación de los laboratorios remotos se evaluará estableciendo los requisitos que deben cumplir los dispositivos para el desarrollo de sistemas embebidos para su adecuación a los diferentes laboratorios remotos implementados. Igual que en el caso anterior se identificarán las diferentes tecnologías soportadas por las implementaciones realizadas.

Igualmente se analizará la interconexión de los componentes desarrollados en cada laboratorio remoto implementado para evaluar la escalabilidad. Se identificarán las limitaciones en cuanto a número de instancias desplegables para cada laboratorio remoto y las causas que comprometen el despliegue de nuevas instancias. Se describirán las implementaciones que disponen de múltiples instancias.

En cuanto a la evaluación de las características generalistas de los laboratorios remotos, hay que recalcar que las implementaciones descritas en el capítulo 4 no se han llevado a cabo únicamente con fines investigativos para la validación de la arquitectura propuesta en este trabajo. Todos los laboratorios remotos descritos en el capítulo 4 han sido implementados con fines educativos, para permitir la experimentación de estudiantes reales sobre tecnologías que permiten el desarrollo de sistemas embebidos. Esto permite evaluar ciertas características generales como la disponibilidad de los laboratorios remotos sobre el registro de las sesiones efectuadas por los estudiantes que han accedido a los laboratorios. Otras características como la universalidad y la conectividad se analizarán identificando los requisitos de dispositivos e instalaciones para permitir el desarrollo de la experimentación desde los laboratorios remotos implementados. Finalmente la integración con plataformas

software de experimentación o aprendizaje se evaluará cuantitativamente identificando los sistemas de gestión de laboratorios remotos, sistemas de gestión del aprendizaje, etc. sobre los que se haya llevado a cabo una integración de los laboratorios remotos implementados.

5.2 Evaluación de los laboratorios remotos implementados en base a las características identificadas

A continuación se hará un repaso de los laboratorios descritos en el capítulo 4. Para cada implementación se analizará en detalle el grado de cumplimiento de cada una de las características identificadas como relevantes en el consumo de laboratorios remotos para experimentación sobre sistemas embebidos haciendo especial hincapié en las tres características señaladas como fundamentales: replicabilidad, escalabilidad y desplegabilidad.

5.2.1 Evaluación del laboratorio remoto WebLab-PIC

El laboratorio remoto WebLab-PIC ha sido implementado como una prueba de concepto para validar la capacidad de implementar la arquitectura propuesta sobre sistemas de bajo coste. Es el único laboratorio remoto en el mundo para permitir la experimentación con sistemas embebidos (microcontroladores), que ha sido implementado integralmente sobre microcontroladores de 8 bits sin requerir de ningún sistema de apoyo adicional.

5.2.1.1 Desplegabilidad en el laboratorio remoto WebLab-PIC

La desplegabilidad es sin lugar a dudas el punto fuerte de este microcontrolador (García-Zubia, Angulo, Hernández-Jayo, & Orduña, 2008). Está implementado íntegramente sobre dos unidades del sistema de desarrollo comercial "PICDEMnet 2" (Microchip Technology, DS51623A, 2006) fácilmente adquirible. Su despliegue en cualquier institución únicamente requiere llevar a cabo cuatro sencillos pasos:

- Programar el firmware. Cada uno de los dos sistemas de desarrollo, monitor y experimento, deben ser programados con el firmware desarrollado. Para ello es necesario un programador compatible con el depurador en circuito (ICD) de la familia PIC18 de Microchip, integrado en el sistema de desarrollo PICDEMnet 2.

- Conectar a la red. El sistema de desarrollo PICDEMnet 2dispone de un conector Ethernet que facilita su conexión a la infraestructura de red. Los firmwares genéricos incluyen un cliente DHCP que facilitan la adquisición de la dirección IP que se utilizará para su acceso. En caso de que la infraestructura de red carezca de servidor DHCP, es posible indicar una dirección IP fija modificando el firmware previamente a su programación.

- Interconectar sistema. El sistema de desarrollo PICDEMnet 2 dispone de un conector de expansión que facilita la conexión entre las dos tarjetas mediante un cable plano.

- Alimentar el sistema. Basta con alimentar una de las dos tarjetas para finalizar el despliegue el sistema.

El total de estos pasos no lleva más de 30 minutos y el coste del sistema supera levemente los 200€. Además, el protocolo Internet Discover implementado en la pila TCP/IP de Microchip permite la detección de las diferentes instancias instaladas en la misma infraestructura de red.

La desplegabilidad de este sistema quedó validada llevando a cabo su despliegue en dos instituciones fuera de la Universidad de Deusto:

1. El laboratorio remoto WebLab-PIC fue desplegado en las instalaciones del Departamento de Dispositivos Electrónicos en la Universidad Politécnica de Budapest, en Hungría durante el congreso "7th Workshop on Microelectronics Education in Europe - (EWME 2008)", durante los días 28 al 30 de mayo del año 2008 (García-Zubia, Angulo, Hernández-Jayo, & Orduña, 2008).
2. El mismo año el sistema WebLab-PIC fue también desplegado en la Universidad de Ciencias Aplicadas de Düsseldorf durante la celebración del congreso "5th Conference on Remote Engineering and Virtual Instrumentation" durante los días 23 al 25 de junio del año 2008 (García-Zubia, Angulo, Hernández, & Orduña, 2008).

En ambos casos el sistema pudo ser accedido y validado inmediatamente tras su conexión a la red general de las instalaciones donde fue desplegado.

5.2.1.2 Escalabilidad en el laboratorio remoto WebLab-PIC

El laboratorio remoto WebLab PIC permite el despliegue tantas instancias como sean soportadas por la red en la que son conectadas. Aunque la configuración por defecto del firmware establece una máscara de red de clase C con un máximo de 254 dispositivos y por tanto 127 instancias, la pila TCP/IP de Microchip sobre la que se

implementa el sistema permite modificar la máscara soportando un número superior de instancias.

Además, la escalabilidad del sistema no afecta la desplegabilidad gracias a la utilización del protocolo Internet Discovery de Microchip. Este protocolo permite localizar todos los sistemas desplegados en la misma red, y generar una página de acceso que facilita al estudiante una lista con el estado de cada instancia, pudiendo acceder a aquellas que se encuentren libres (Figura 4.10).

5.2.1.3 Replicabilidad en el laboratorio remoto WebLab-PIC

Además de por motivos de limitación del espacio de la memoria de programa, la implementación de los diferentes componentes identificados en la arquitectura propuesta sobre dos microcontroladores diferentes responde a la demanda de replicabilidad de los laboratorios remotos para experimentación sobre tecnologías que permiten el desarrollo de sistemas embebidos. De esta forma todos los componentes independientes de la tecnología de experimentación han sido implementados sobe el microcontrolador denominado "monitor", mientras que los componentes dependientes (servidor de grabación y plataforma de experimentación) han sido implementados sobre el microcontrolador denominado "experimento" (Figura 4.3).

Esta división permite sustituir la plataforma de experimentación y consiguientemente el servidor de grabación, sin tener que modificar el resto de componentes. La replicabilidad en el laboratorio WebLab-PIC permite su adecuación a cualquier dispositivo de tecnología embebida que permita su programación a través de protocolo TFTP.

Un ejemplo que prueba la replicabilidad de este laboratorio es la adecuación del mismo a la plataforma Arduino. El desarrollo de un bootloader TFTP para el microcontrolador ATmega328P de Atmel (Atmel, 2014), compatible con la plataforma Arduino que permite la reprogramación de la memoria de programa desde ficheros codificados en formato Intel Hex (Hazarinov & Perotto, 2012) ha permitido alterar la plataforma de experimentación a una Arduino Ethernet Rev. 3 (Arduino, 2014) sin alterar ningún componente adicional del laboratorio remoto.

5.2.1.4 Características generales a los laboratorios remotos en el sistema WebLab-PIC

Validado el cumplimiento de las tres características identificadas como fundamentales en el desarrollo de sistemas para experimentación remota con sistemas

embebidos, a continuación se evalúa el cumplimiento de las características generales que caracterizan la eficacia de los laboratorios remotos en el WebLab-PIC:

- *Integración con otras plataformas de aprendizaje o experimentación*. La implementación del laboratorio remoto WebLab-PIC se ha centrado fundamentalmente en facilitar al máximo el despliegue del sistema. Por este motivo todos los componentes del laboratorio se han desarrollado de manera independiente (standalone), para posibilitar el acceso al laboratorio mediante su simple conexión a la red de la infraestructura donde se despliega. Los conocimientos necesarios para instalar los sistemas de gestión de laboratorios remotos (WebLab-Deusto, 2014) o plataformas de aprendizaje (Johnson & Miller, 2013) complican el despliegue y comprometen la consecución de los objetivos marcados, no habiendo sido tenida en cuenta la integración del laboratorio remoto WebLab-PIC con este tipo de plataformas software.

- *Disponibilidad*. La disponibilidad del laboratorio remoto es administrada directamente por el administrador encargado de su despliegue. WebLab-PIC dispone de un sistema de autenticación independiente, permitiendo al administrador gestionar las credenciales de los usuarios autorizados. Las limitaciones de rendimiento de la plataforma sobre la que se ha desplegado el servidor de administración han impedido el desarrollo de un sistema avanzado de colas o reservas.

- *Conectividad*. Todo el interfaz de usuario del laboratorio remoto WebLab-PIC está desarrollado sobre un servidor Web sobre el puerto TCP 80 ya que no dispone de conexión segura. Aunque la comunicación entre el servidor del laboratorio y el servidor de grabación se lleva a cabo mediante el protocolo TFTP mediante el puerto UDP 69, esta se desarrolla entre nodos conectados en la misma red sin generar un problema de conectividad.

- *Universalidad*. Como se ha comentado anteriormente, todo el interfaz de usuario de este laboratorio es accesible desde un navegador HTML. Esto garantiza la accesibilidad del laboratorio desde dispositivos móviles para el desarrollo de las fases de programación y validación de en la experimentación con sistemas embebidos. Lamentablemente, el desarrollo de las fases de edición y compilación requieren un entorno de desarrollo compatible con la familia de microcontroladores PIC18 de Microchip. En este sentido el bootloader desarrollado para la implementación del servidor de grabación permite al estudiante seleccionar cualquier entorno compatible sin requerir llevar a cabo ninguna configuración específica para el laboratorio remoto.

5.2.2 Evaluación del laboratorio remoto WebLab-Robot

La implementación del laboratorio remoto WebLab-Robot se llevó a cabo tras la evaluación del sistema WebLab-PIC para validar la capacidad de integración de los laboratorios en plataformas software de aprendizaje o experimentación. A continuación se evalúa el grado de cumplimiento de este laboratorio remoto respecto a las características establecidas.

5.2.2.1 Desplegabilidad en el laboratorio remoto WebLab-robot

Todos los componentes hardware utilizados en la implementación del laboratorio remoto WebLab-Robot son sistemas de desarrollo comerciales fácilmente adquiribles por cualquier persona o institución que desee desplegar una instancia del laboratorio (Ingeniería de Microsistemas Programados, 2010) (Ingeniería de Microsistemas Programados, 2008). Asimismo, instrucciones detalladas sobre su implementación están publicadas con acceso abierto en el portal WebLab-Deusto.

Desde el punto de vista software, los componentes servidor de administración, servidor del laboratorio y cliente han sido desarrollados aprovechando los recursos proporcionados por el sistema de gestión de laboratorios remotos WebLab-Deusto. En este sentido, la comunidad que colabora con el desarrollo de este RLMS trabaja en facilitar la instalación de esta plataforma software, exigiendo simplemente conocimientos medios de administración de sistemas operativos Linux o Microsoft Windows (Orduña, y otros, 2011).

La desplegabilidad de este laboratorio se ha validado mediante la instalación de dos instancias de este laboratorio en dos instalaciones ajenas a sus desarrolladores:

- La participación de la Universidad de Deusto en el proyecto "Building an Ecology of Online Laboratories" (Award#: 1132813) (Orduña, y otros, 2012) subvencionado por la NSF (National Science Foundation) en colaboración con la Universidad Nacional a Distancia (UNED) y con el Instituto Tecnológico de Massachussetts (MIT) permitió el despliegue de una instancia de este laboratorio en esta última institución dentro de las instalaciones del Centro para Iniciativas Computacionales Educativas (CECI).

- Recientemente y gracias a la participación de la Universidad de Deusto en el proyecto iCo-op subvencionado por el Programa Tempus de la Comunidad Europea (0278-TEMPUS-1-2012-1-DE-Tempus-JPH), y concedido a un total de 22 socios formados por universidades y empresas europeas (TU-Ilmenau.de/ics, 2013), se ha llevado a cabo la instalación de otra instancia de este laboratorio en la Universidad Estatal Shota Rustaveli en Batumi, Georgia.

El despliegue de estas dos instancias por instituciones ajenas a los desarrolladores, valida la desplegabilidad de este laboratorio.

5.2.2.2 Escalabilidad en el laboratorio remoto WebLab-Robot

El cumplimiento de las especificaciones definidas por la arquitectura abierta para el despliegue de laboratorios remotos con experimentación sobre sistemas embebidos unido a la integración del laboratorio remoto WebLab-Robot en el sistema de gestión de laboratorios remotos Weblab-Deusto garantizan la escalabilidad del laboratorio. Por un lado la implementación independiente de los componentes que conforman el laboratorio permite el despliegue paralelo de tantas instancias como nuevos nodos admita la red de la infraestructura sobre la que se despliegue. Por otro lado WebLab-Deusto ofrece balanceo de carga entre instancias idénticas facilitando el consumo de las mismas por los estudiantes y proporcionando un mecanismo basado en colas para el establecimiento de sesiones de experimentación.

El cumplimiento de la escalabilidad de este laboratorio puede validarse empíricamente desde el portal de experimentación remota desplegado en la Universidad Estatal Shota Rustaveli en Batumi, Georgia. Este portal permite a los estudiantes de esta institución acceder indistintamente a las instancias de este laboratorio desplegadas en Bilbao y Batumi. Los estudiantes pueden acceder específicamente a una de ellas o, al tratarse de experimentos idénticos, solicitar el acceso a la primera instancia que quede libre (Figura 5.1).

Figura 5.1. Portal de experimentación remota de la Universidad Shota Rustaveli en Batumi, Georgia.

5.2.2.3 Replicabilidad en el laboratorio remoto WebLab-Deusto

El laboratorio remoto WebLab-Robot puede adecuarse fácilmente a cualquier tecnología para el desarrollo de sistemas embebidos que permita la reprogramación de su memoria de programa mediante un bootloader a través de UART. Esto incluye a la mayoría de microcontroladores y sistemas en un chip (SoC)(Gabriel & Ovidiu, 2003).

La prueba de replicabilidad de este laboratorio se ha llevado a cabo desplegando una nueva instancia de este laboratorio basada en la plataforma Arduino UNO. El laboratorio remoto WebLab-Romie ha sido desplegado sobre la base del WebLab-Robot sustituyendo únicamente la plataforma de experimentación y el servidor de grabación asociado (Iturrate, Angulo, & Orduña, 2013) (Figura 5.2).

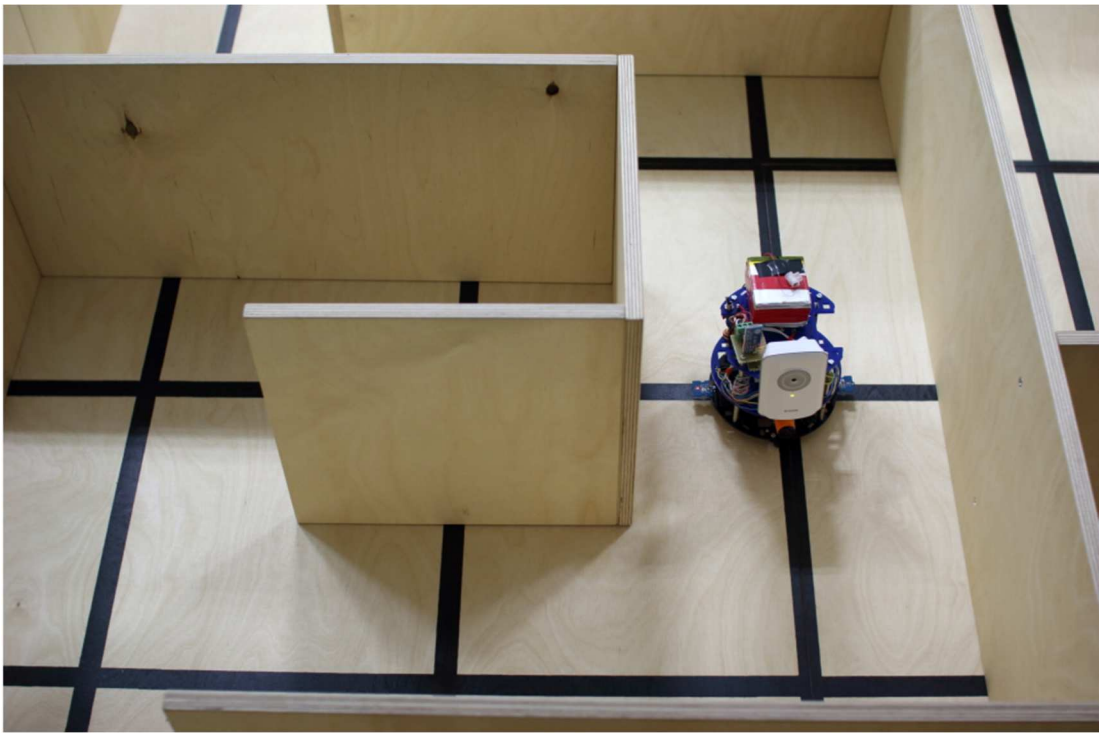

Figura 5.2. Imagen del laboratorio remoto Romie, desplegado en el Museo Politécnico Nacional de Sofía, en Bulgaria.

5.2.2.4 Características generales a los laboratorios remotos en el sistema WebLab-Robot

El grado de cumplimiento del resto de características genéricas a los laboratorios remotos del sistema WebLab-Robot se muestran a continuación:

- *Integración con otras plataformas de aprendizaje o experimentación*. El laboratorio remoto WebLab-Robot está completamente integrado en el sistema de gestión de laboratorios remotos WebLab-Deusto. Por su parte, el RLMS WebLab-Deusto participa en la iniciativa Gateway4Labs (Orduña, y otros, 2013) (Orduña, y otros, 2014) dirigida a la integración de múltiples laboratorios remotos en diferentes entornos de aprendizaje digitales, tales como sistemas de gestión de aprendizaje (LMS), Sistemas de Gestión de Contenidos (CMS) o entornos de aprendizaje personales (Figura 5.3). Mediante la herramienta Labmanager surgida de esta iniciativa, todos los laboratorios remotos integrados en el RLMS WebLab-Deusto pueden federarse desde otros sistemas de gestión de laboratorios remotos (iLabs, VirtualLabs, Concord, etc.) así como accederse desde las principales herramientas de aprendizaje de aprendizaje (Moodle, Sakai, .LRN etc.) (gateway4labs team, 2012) (Orduña, y otros, 2013) (Orduña, y otros, 2014).

- *Disponibilidad*. El sistema WebLab-Robot es uno de los laboratorios remotos publicados en el portal WebLab-Deusto que permite su acceso a invitados. Este sistema fue publicado en el año 2011 y desde entonces acumula más de 7000 sesiones desde más de 75 países (Figura 5.4).

- *Conectividad*. Su integración con WebLab-Deusto permite acceder al laboratorio desde un servidor Web. Esto garantiza la conectividad del sistema sin necesitar ninguna configuración específica en la infraestructura en la que se despliegue.

- *Universalidad*. todo el interfaz de usuario de este laboratorio es accesible desde un navegador HTML. Para el desarrollo de las fases de edición y compilación se requiere un entorno de desarrollo compatible con la familia de microcontroladores PIC18 de Microchip. Desde WebLab-Deusto se ofrecen ficheros linker script compatibles con los compiladores oficiales de Microchip, MPLAB-C18 y XC-8, que permiten el desarrollo de sistemas sin contemplar las configuraciones específicas requeridas por el bootloader que soporta el servidor de grabación.

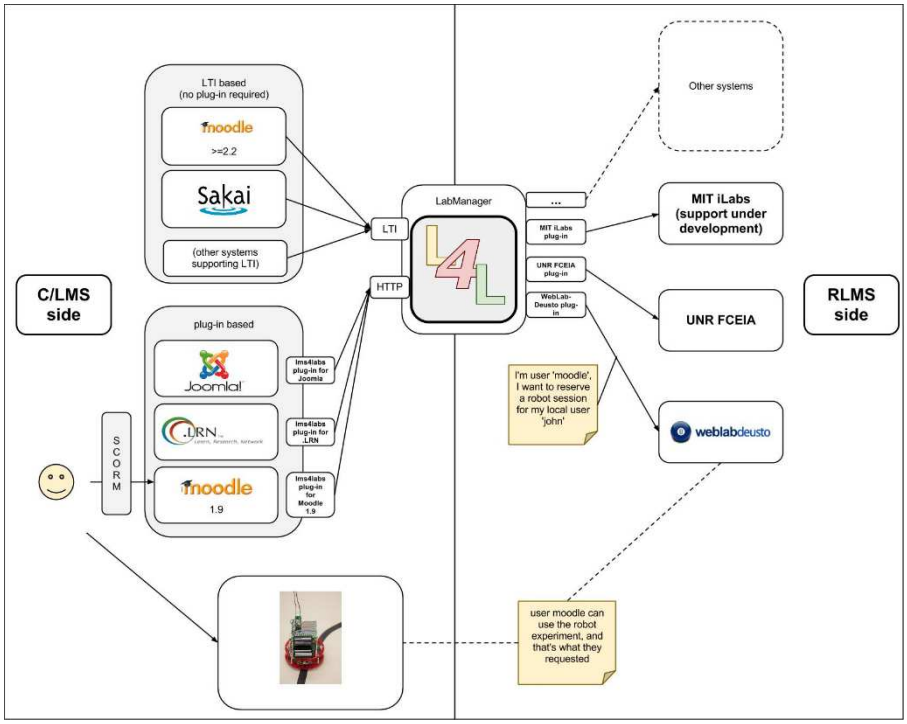

Figura 5.3. Arquitectura general de la iniciativa Gateway4Labs.

Figura 5.4. Mapa representativo de los países desde los que se ha accedido al laboratorio remoto WebLab-Robot.

5.2.3 Evaluación de los laboratorios remotos implementados mediante la plataforma WebLab-Box

El objetivo de las múltiples implementaciones de laboratorios remotos sobre diferentes tecnologías para el desarrollo de sistemas embebidos llevadas a cabo sobre la plataforma WebLab-Box ha sido validar la arquitectura propuesta en base a las tres características definidas como fundamentales en los laboratorios remotos para la provisión de esta experimentación.

5.2.3.1 Desplegabilidad en los laboratorios remotos implementados sobre la plataforma WebLab-Box

Uno de los principales problemas que ha frenado históricamente la democratización de los laboratorios remotos ha sido el autoconsumo de los mismos por parte de los propios desarrolladores. El objetivo principal de la plataforma WebLab-Box es posibilitar el despliegue de laboratorios remotos para experimentación con tecnologías de desarrollo de sistemas embebidos a todos los profesionales dedicados a la enseñanza de las mismas, sin requerir experiencia en el diseño e implementación de laboratorios remotos.

La plataforma WebLab-Box ha sido diseñada en colaboración con una empresa para garantizar la sostenibilidad del proyecto. La empresa Elson Electrónica S.A. dispone de capacidad de fabricación y comercialización de la plataforma básica WebLab-Box para el despliegue de laboratorios remotos para experimentación sobre tecnologías embebidas.

Toda la información requerida para desplegar laboratorios remotos a medida basados en la plataforma WebLab-Box está accesible desde el portal WebLab-Deusto.

La desplegabilidad de esta plataforma ha quedado validada mediante dos despliegues llevados a cabo fuera de la Universidad de Deusto.

- Una instancia del WebLab-Box para experimentación con microcontroladores ha sido desplegada en el Instituto Tecnológico de Massachussetts (MIT) dentro del proyecto "Building an Ecology of Online Laboratories" (Award#: 1132813) (Orduña P. , y otros, 2012).
- Una instancia el WebLab-Box para experimentación con dispositivos programables ha sido desplegada en la Universidad Estatal Shota Rustaveli en Batumi, Georgia, subvencionada por la Comunidad Europea bajo el programa Tempus dentro del proyecto "Industrial Cooperation and Creative Engineering Education based on Remote Engineering and Virtual

Instrumentation (iCo-op)" (0278-TEMPUS-1-2012-1-DE-Tempus-JPH) (TU-Ilmenau.de/ics, 2013).

5.2.3.2 Escalabilidad de los laboratorios remotos desplegados sobre la plataforma WebLab-Box

Como principal solución al acceso concurrente a laboratorios remotos para la experimentación con tecnologías para el desarrollo de sistemas embebidos, la búsqueda de escalabilidad fue una característica fundamental en el diseño de la plataforma WebLab-Box.

La escalabilidad de esta plataforma queda validada mediante la convivencia de 5 instancias paralelas en las instalaciones del laboratorio WebLab-Deusto.

Estas 5 instancias se encuentran integradas en el sistema de gestión de laboratorios remotos WebLab-Deusto, que proporciona el balanceo de carga que posibilita el consumo de las múltiples instancias de una forma transparente para los estudiantes. Cuando un estudiante solicita reservar un laboratorio para el desarrollo de un experimento, el RLMS WebLab-Deusto encola la petición hasta la liberación de la primera instancia compatible con la tecnología de experimentación requerida (Orduña, y otros, 2011).

La federación entre laboratorios remotos de diferentes instituciones (Orduña, 2013) permite llevar la escalabilidad a un nivel superior. Por ejemplo los alumnos de la Universidad Estatal Shota Rustaveli, que disponen de una única instancia de laboratorio remoto para experimentación con dispositivos programables basada en la plataforma WebLab-Box en sus instalaciones, acceden transparentemente a las instancias compatibles desplegadas en la Universidad de Deusto en caso de encontrarse libres.

5.2.3.3 Replicabilidad de los laboratorios remotos basados en la plataforma WebLab-Box

La plataforma WebLab-Box, basada en la arquitectura abierta para el despliegue de laboratorios remotos sobre de sistemas embebidos, ha sido diseñada para permitir la experimentación con cualquier tecnología.

Laboratorios remotos basados en múltiples tecnologías han sido implementados para la validación de esta característica:

- Microcontroladores PIC de la familia 16F.
- Microcontroladores PIC de la familia 18F.

- Dispositivos lógicos programables complejos (CPLD) de la familia XC95XX
- Field Programmable Gate Array (FPGA) de la familia Spartan-3

Actualmente dos laboratorios remotos basados en microcontroladores ARM Cortex M y sistemas en un chip (SoC) Xilinx Zinq7000 se encuentran en desarrollo.

5.2.3.4 Características generales a los laboratorios remotos en los sistemas implementados bajo la plataforma WebLab-Box

A continuación se describe el cumplimento del resto de características identificadas como relevantes en el rendimiento de laboratorios remotos sobre el conjunto de sistemas implementados bajo la plataforma WebLab-Box.

- *Integración con otras plataformas de experimentación o aprendizaje:* Aunque se dispone de un software "standalone" para el funcionamiento integral e independiente de los laboratorios basados en esta plataforma (García-Zubia, y otros, 2010), la plataforma está diseñada para su integración con el RLMS WebLab-Deusto, como ha quedado probado en las múltiples instancias de los laboratorios WebLab-FPGA, WebLab-CPLD y WebLab-PIC2. La herramienta Labmanager desarrollada dentro de la iniciativa Gateway4Labs (gateway4labs team, 2012) permite su integración con las principales herramientas educativas y sistemas de gestión de laboratorios remotos (Figura 5.3).

- *Disponibilidad:* Las múltiples instancias de laboratorios remotos basadas en la plataforma WebLab-Box publicadas en el portal de experimentación WebLab-Deusto se encuentran disponibles todos los días de año a cualquier hora del día. Actualmente basta solicitar acceso a cualquiera de esos laboratorios para conseguir las credenciales que dan acceso a la experimentación. Los laboratorios basados en la plataforma WebLab-Box han soportado más de 40.000 sesiones de experimentación levadas a cabo desde 88 países. La Figura 5.5 muestra los puntos del mapa que han accedido a los laboratorios remotos bajo esta plataforma.

- *Conectividad:* Como se ha indicado anteriormente, tanto los laboratorios basados en la plataforma WebLab-Box desplegados utilizando la versión standalone como los integrados en el RLMS WebLab-Deusto permiten la conectividad en cualquier instalación sin requisitios especiales. Ambas versiones son íntegramente accesibles desde un navegador de Internet por los puertos habituales (80, 443).

- *Universalidad:* Aunque esta característica es dependiente de la tecnología, ya que existen tecnologías para el desarrollo de sistemas embebidas que únicamente soportan su sintetizado/compilación desde herramientas propietarias (Xilinx, 2009), la arquitectura propuesta e implementada en la plataforma WebLab-Box dispone de mecanismos para el desarrollo del esta etapa del diseño desde el servidor de grabación permitiendo llevar a cabo una experimentación integral desde cualquier dispositivo con un navegador web. Las dos instancias de los laboratorios WebLab-CPLD publicadas en el portal de experimentación WebLab-Deusto bajo la plataforma WebLab-Box, permiten la subida por parte del estudiante de un fichero binario, pero también admiten archivos fuente en lenguaje de definición de hardware VHDL, realizando la síntesis, adecuación y rutado del diseño en el propio servidor de grabación antes de ejecutar la programación del dispositivo.

Figura 5.5. Mapa representativo de los países desde los que se ha accedido a los laboratorios remotos implementados bajo la plataforma WebLab-Box.

5.2.4 Evaluación del laboratorio remoto WebLab-Elevator

Es necesario recordar que una vez validada la replicabilidad de los laboratorios remotos basados en la plataforma WebLab-Box respecto a las diferentes tecnologías para el desarrollo de sistemas embebidas, el propósito de la implementación del laboratorio remoto WebLab-Elevator es validar la replicabilidad de la arquitectura abierta para el despliegue de laboratorios remotos sobre sistemas embebidos desde el

punto de vista de la plataforma de experimentación, atendiendo a la naturaleza del propio experimento y del sistema a controlar desde el sistema embebido.

Las limitaciones físicas y eléctricas de la plataforma WebLab-Box, diseñadas para soportar sistemas de desarrollo con propósito didáctico, impiden el control de equipamiento industrial debido al gran tamaño y a las características eléctricas que suelen presentar las maquetas didácticas automatizadas. El WebLab-Elevator presenta una implementación idéntica a la soportada por la plataforma WebLab-Box sobre la que se han incluido una fuente de alimentación industrial y sistemas de adecuación de niveles para la integración con una maqueta didáctica comercial.

En este sentido, este laboratorio presenta el mismo grado de cumplimiento de las características indicado en el apartado 5.2.3 para los laboratorios remotos basados en la plataforma WebLab-Box, pero además permite validar la arquitectura propuesta desde el punto de vista de la replicabilidad ante el sistema a controlar (Figura 5.6).

Figura 5.6. Laboratorio remoto WebLab-Elevator.

En estos momentos el laboratorio remoto WebLab-Elevator permite el control de la maqueta desde una FPGA Spartan-3. Actualmente el equipo WebLab-Deusto está trabajando en el diseño de una matriz de conmutación que permitirá al estudiante

seleccionar entre distintas tecnologías (FPGA, MCU, PLC, etc.) para experimentar con el control de la maqueta del ascensor de tres plantas.

5.3 Evaluación de los requisitos de la arquitectura propuesta sobre las implementaciones

Como se ha explicado en la metodología, tras evaluar el cumplimiento de las principales características en cada uno de los laboratorios implementados bajo la arquitectura propuesta, se llevará a cabo una evaluación de cada característica respecto al total de los laboratorios implementados.

El propósito de esta redundancia en la evaluación es validar cualitativamente el comportamiento de los laboratorios implementados siguiendo la arquitectura planteada en base a las características establecidas en el estado del arte como relevantes para el rendimiento de un laboratorio remoto que permite la experimentación con las diferentes tecnologías que se utilizan en el desarrollo de sistemas embebidos.

5.3.1 Desplegabilidad

La desplegabilidad de un laboratorio remoto depende directamente de la accesibilidad de los componentes hardware/software utilizados en su implementación, en conjunción con una documentación que describa la metodología a seguir para su instalación. En este sentido la arquitectura propuesta determina la disposición de los componentes que conforman el laboratorio remoto y establece la funcionalidad independiente de cada componente. Esto facilita la adecuación de los laboratorios remotos ya implementados a los requisitos específicos de las organizaciones que llevan a cabo nuevos despliegues, permitiendo la construcción de sistemas que van más allá de meras réplicas y proponen nuevos experimentos remotos.

La desplegabilidad de la arquitectura ha quedado validada con los múltiples despliegues llevados a cabo en instalaciones ajenas a la Universidad de Deusto. Los siguientes laboratorios remotos han sido desplegados siguiendo la arquitectura propuesta en el capítulo 3:

1. El laboratorio remoto para experimentación con el microcontroladores de 8 bits desplegado en las instalaciones del Departamento de Dispositivos Electrónicos en la Universidad Politécnica de Budapest, en Hungría durante el congreso "7th Workshop on Microelectronics Education in Europe - (EWME

2008)", durante los días 28 al 30 de mayo del año 2008 (García-Zubia, Angulo, Hernández-Jayo, & Orduña, 2008).

2. Laboratorio remoto para experimentación con el microcontroladores de 8 bits desplegado en la Universidad de Ciencias Aplicadas de Düsseldorf durante la celebración del congreso "5th Conference on Remote Engineering and Virtual Instrumentation" durante los días 23 al 25 de junio del año 2008 (García-Zubia, Angulo, Hernández, & Orduña, 2008).
3. Laboratorio remoto para experimentación con robots móviles desplegado en el Instituto Tecnológico de Massachussetts (MIT) dentro del proyecto "Building an Ecology of Online Laboratories" (Award#: 1132813) (Orduña, y otros, 2012) subvencionado por la NSF (National Science Foundation).
4. Laboratorio remoto para experimentación con robots móviles desplegado en la Universidad Estatal Shota Rustaveli en Batumi, Georgia dentro del proyecto "Industrial Cooperation and Creative Engineering Education based on Remote Engineering and Virtual Instrumentation (iCo-op)" subvencionado por el Programa Tempus de la Comunidad Europea, y concedido a un total de 22 socios formados por universidades y empresas europeas (0278-TEMPUS-1-2012-1-DE-Tempus-JPH) (TU-Ilmenau.de/ics, 2013).
5. Laboratorio para experimentación con microcontroladores desplegado sobre la plataforma WebLab-Box en el Instituto Tecnológico de Massachussetts (MIT) dentro del proyecto "Building an Ecology of Online Laboratories" (Award#: 1132813) (Orduña P. , y otros, 2012).
6. Laboratorio para experimentación con dispositivos programables desplegado sobre la plataforma WebLab-Box en la Universidad Estatal Shota Rustaveli en Batumi, Georgia, subvencionada por la Comunidad Europea bajo el programa Tempus dentro del proyecto "Industrial Cooperation and Creative Engineering Education based on Remote Engineering and Virtual Instrumentation (iCo-op)" (0278-TEMPUS-1-2012-1-DE-Tempus-JPH) (TU-Ilmenau.de/ics, 2013).

5.3.2 Escalabilidad

Uno de los fundamentos en los que se sustenta la arquitectura propuesta es la interconexión de los diferentes componentes del laboratorio remoto sobre los que se sustenta la experimentación remota a través de una red local. Esto permite identificar a cada subsistema por medio de una dirección única, permitiendo discriminar las diferentes instancias de un mismo laboratorio remoto en base a las direcciones de sus componentes. De esta forma se garantiza la escalabilidad del sistema.

El laboratorio WebLab-Deusto de experimentación remota de la Universidad de Deusto dispone en su catálogo de 10 instancias de laboratorios remotos para experimentación con tecnologías embebidas implementados mediante la arquitectura:

1. Dos instancias del laboratorio WebLab-PIC (apartado 4.1).
2. Laboratorio WebLab-Robot (apartado 4.2).
3. Dos instancias del laboratorio WebLab-FPGA bajo la plataforma WebLab-Box (apartado 4.3).
4. Dos instancias del laboratorio WebLab-CPLD bajo la plataforma WebLab-Box (apartado 4.3).
5. Laboratorio WebLab-PIC (apartado 4.3).
6. Laboratorio WebLab-MCU (apartado 4.3).
7. Laboratorio WebLab-Elevator (apartado 4.4).

Algunas de estas instancias se encuentran integradas con el sistema de gestión de laboratorios remotos WebLab-Deusto, mientras que otras disponen de un sistema standalone que permite su consumo independiente por parte del estudiante.

La convivencia de todas estas instancias valida la escalabilidad del sistema.

5.3.3 Replicabilidad

La arquitectura abierta para el despliegue de laboratorios en sistemas embebidos desacopla la funcionalidad de los elementos que físicamente ejecutan la experimentación en tres subsistemas independientes:

- Servidor de grabación.
- Servidor de interacción.
- Plataforma de experimentación.

De esta forma se consigue independizar la experimentación ofrecida de la tecnología sobre la que se desarrolla. Esto optimiza la replicabilidad de los laboratorios remotos, minimizando los componentes que deben ser alterados ante una variación en los dispositivos que soportan la experimentación.

La arquitectura abierta para la desplegabilidad de laboratorios remotos con experimentación sobre tecnologías para el desarrollo de sistemas embebidos ha sido validada mediante la implementación de laboratorios remotos que permiten la experimentación sobre los siguientes dispositivos:

- Microcontrolador PIC18F97J60 (LR WebLab-PIC).

- Microcontrolador PIC18F4550 (LR WebLab-Robot).
- Microcontrolador PIC18F45k22 (LR WebLab-MCU).
- Microcontrolador PIC16F877 (LR WebLab-MCU).
- FPGA Spartan 6 (LR WebLab-FPGA y LR WebLab-Elevator).
- CPLD XC95X72 (LR WebLab-CPLD).
- Plataforma Arduino UNO (LR Romie).

El despliegue de laboratorios basados en la arquitectura propuesta que permiten la experimentación con múltiples tecnologías permite validar la replicabilidad de la misma.

5.3.4 Otras características relevantes

Aunque el diseño de la arquitectura abierta para el despliegue de laboratorios remotos sobre tecnologías de desarrollo de sistemas embebidos se ha fundamentado en la consecución de desplegabilidad, escalabilidad y replicabilidad, las implementaciones llevadas a cabo siguiendo las especificaciones establecidas en la definición de la arquitectura propuesta permiten validar el cumplimiento de la misma respecto a otras características generales a los laboratorios remotos de distinta naturaleza.

5.3.4.1 Integración con otras plataformas de experimentación o aprendizaje

Los laboratorios remotos implementados sobre la arquitectura propuesta han sido integrados en multitud de plataformas de experimentación o aprendizaje. En ocasiones la integración se ha llevado a cabo durante el propio despliegue, mientras que en otras ocasiones se ha logrado mediante la federación entre las propias plataformas (Orduña, 2013).

El resumen de plataformas en las que los laboratorios remotos desplegados bajo las especificaciones de la arquitectura propuesta es el siguiente:

- Sistema de Gestión de Laboratorios Remotos WebLab-Deusto (Orduña, y otros, 2011).
- Sistema de Gestión de Laboratorios Remotos iLab (Orduna, Rodriguez-Gil, & Lopez-de-Ipina, 2012).
- Sistema de Gestión de Laboratorios Remotos Labshare Sahara (Orduna, Rodriguez-Gil, & Lopez-de-Ipina, 2012).
- Sistema de gestión del aprendizaje Moodle (García-Zubia, Orduña, Irurzun, Angulo, & Hernandez-Jayo, 2009).

Además, el desarrollo de la herramienta Labmanager por parte de la iniciativa Gateway4Labs proporciona los mecanismos que facilitan la interconexión entre laboratorios remotos y herramientas de aprendizaje (gateway4labs team, 2012). La inclusión del sistema de gestión de laboratorios remotos WebLab-Deusto en esta herramienta garantiza la integración con las principales plataformas de experimentación y aprendizaje.

Por último, el despliegue de laboratorios remotos sobre plataformas standalone (WebLab-PIC y WebLab-MCU) valida el funcionamiento independiente de los laboratorios remotos implementados siguiendo las especificaciones de la arquitectura.

5.3.4.2 Disponibilidad

Todos los laboratorios remotos implementados bajo la arquitectura disponen de los mecanismos de autenticación y gestión de sesiones que garantizan la disponibilidad de los mismos para los estudiantes.

Además el registro de actividad del RLMS WebLab-Deusto permite contabilizar las sesiones, usuarios y ubicaciones de acceso de los laboratorios remotos integrados. La Figura 5.7 muestra el número de sesiones totales por cada año del conjunto de laboratorios remotos desplegados sobre el RLMS WebLab-Deusto, siguiendo la arquitectura propuesta:

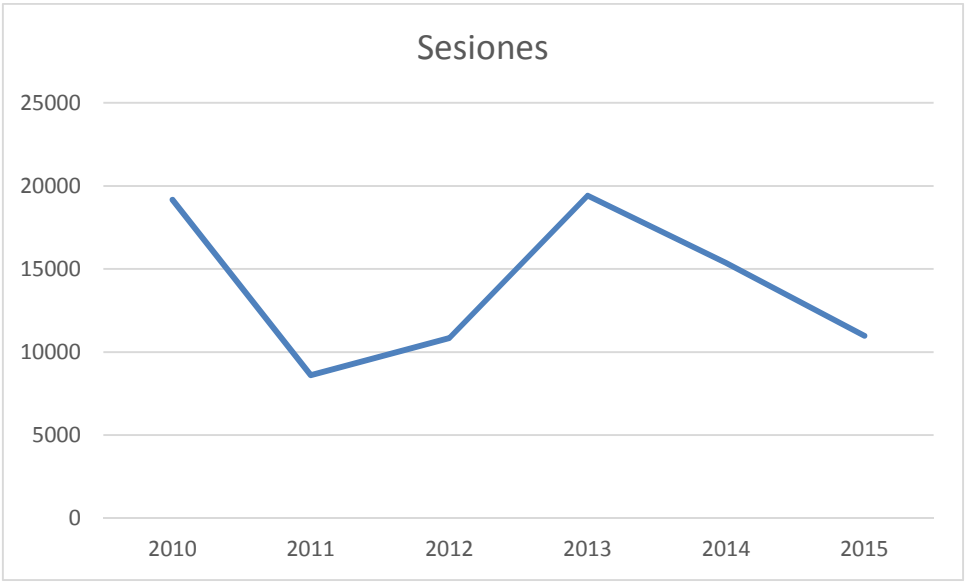

Figura 5.7. Número de sesiones por año en el conjunto de laboratorios implementados siguiendo la arquitectura abierta para el despliegue de laboratorios remotos para experimentación sobre tecnologías de desarrollo de sistemas embebidos integrados en el RLMS WebLab-Deusto.

La Figura 5.8 muestra el número de sesiones llevadas a cabo en los laboratorios remotos implementados siguiendo las especificaciones de la arquitectura propuesta que están integrados en el sistema de gestión de laboratorios remotos WebLab-Deusto desde el año 2012.

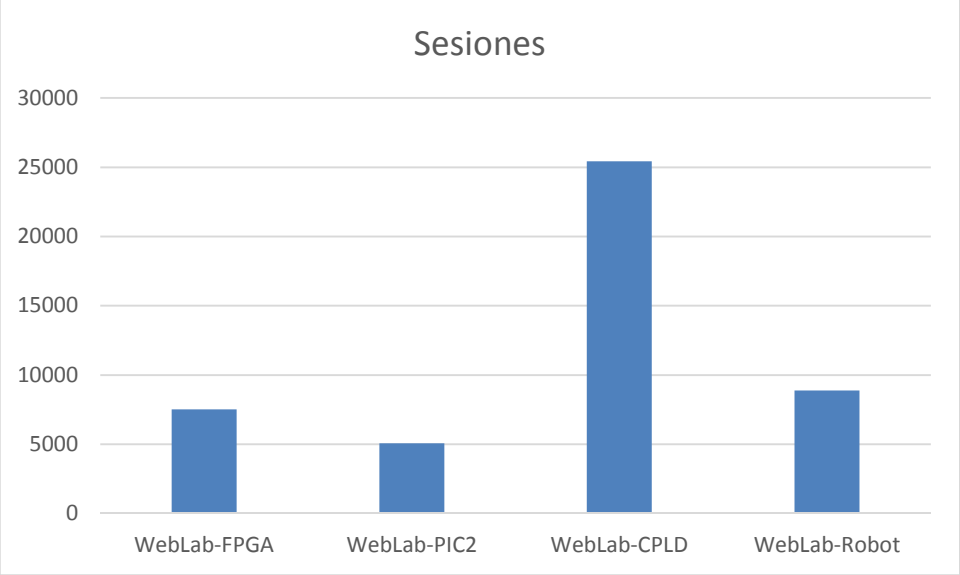

Figura 5.8. Sesiones de experimentación llevadas a cabo en los laboratorios remotos implementados siguiendo la arquitectura abierta para el despliegue de laboratorios remotos para experimentación sobre tecnologías de desarrollo de sistemas embebidos integrados en el RLMS WebLab-Deusto.

5.3.4.3 Conectividad

Todos los laboratorios cuya implementación se describe en el capítulo 4 no requieren ninguna configuración específica en la infraestructura de red donde son desplegados. Toda la comunicación con el cliente se lleva cabo a través de un servidor HTML a través de los puertos estándar.

La muestra el análisis de depuración web durante una sesión de experimentación en el laboratorio remoto WebLab-FPGA desde el navegador Firefox V39.0 capturada utilizando el proxy de depuración web Telerik fiddler (Figura 5.9).

#	Result	Protocol	Host	URL	Body	Caching	Content-Type	Process
1	200	HTTP	Tunnel to	weblab.deusto.es:443	0			firefox:...
2	200	HTTP	Tunnel to	weblab.deusto.es:443	0			firefox:...
3	200	HTTP	Tunnel to	weblab.deusto.es:443	0			firefox:...
4	200	HTTP	Tunnel to	cams.weblab.deusto.es:443	0			firefox:...
5	200	HTTP	Tunnel to	weblab.deusto.es:443	0			firefox:...

Figura 5.9. Análisis de depuración web de una sesión de experimentación completa llevada a cabo sobre el laboratorio remoto WebLab-Deusto.

5.3.4.4 Universalidad

La universalidad aplicada a laboratorios remotos refleja la variedad de dispositivos desde los cuales es posible llevar a cabo una sesión de experimentación.

Como se ha comentado con anterioridad, en el caso específico de los laboratorios remotos que permiten la experimentación sobre tecnologías de desarrollo de sistemas embebidos, esta característica está en ocasiones comprometida por la propia tecnología de experimentación.

Todos los fabricantes de dispositivos embebidos (MCU, FPGA, CPLD, DSP, etc.) proporcionan las herramientas software que permiten llevar a cabo la fase de compilación o síntesis. En ocasiones, proporcionan la información suficiente que posibilita a terceros el desarrollo de herramientas compatibles. Esto permite la existencia de herramientas libres que permiten la compilación de ciertos microcontroladores e incluso la síntesis de algún dispositivo programable. Otras veces la única posibilidad para llevar a cabo esa fase de la experimentación es la utilización de las herramientas oficiales, que suelen estar sujetas a los términos de las licencias correspondientes. En esos casos, la universalidad de los laboratorios remotos se compromete no por la arquitectura del mismo sino por la propia tecnología de experimentación.

Para la validación de esta característica en el laboratorio WebLab-CPLD se han integrado los procesos de síntesis, mapeado y rutado del dispositivo dentro del servidor de grabación. Los estudiantes pueden enviar el fichero fuente que contiene el experimento en lenguaje VHDL y el propio servidor de grabación se encarga de llevar a cabo todo el proceso de sintetizado, mapeado y rutado, previamente a ejecutar la programación del dispositivo.

Este sistema mejora la universalidad del experimento puesto que únicamente es necesario un dispositivo con un navegador web para ejecutar todo el proceso de experimentación. Esto permite llevar a cabo experimentación con tecnologías de desarrollo de sistemas embebidos mediante sistemas móviles (Figura 5.10), chromebooks e incluso desde cualquier ordenador de acceso público sin necesidad de instalar ninguna aplicación.

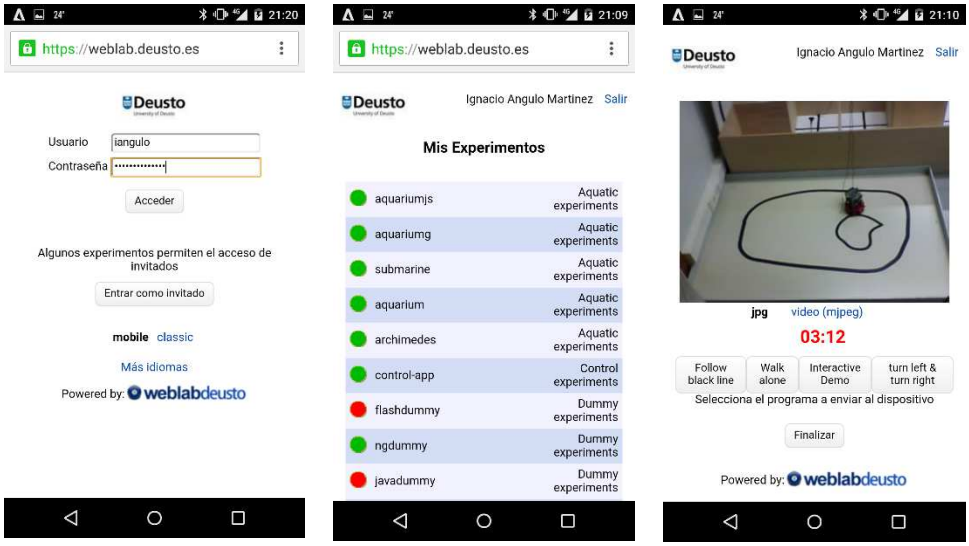

Figura 5.10. Capturas de pantalla tomadas en sesiones de experimentación realizadas desde dispositivos móviles.

En contrapartida el elevado tiempo que conlleva la ejecución de los pasos de sintetizado, mapeo y rutado elevan los tiempos necesarios para cada sesión comprometiendo el acceso al laboratorio por parte de grupos elevados de estudiantes.

5.4 Evaluación subjetiva de los laboratorios remotos

Aunque la evaluación subjetiva por parte de los usuarios de los laboratorios remotos implementados no es el principal objetivo en la validación de la arquitectura propuesta, enriquece desde un distinto punto de vista dicha evaluación.

La Tabla 5.1 muestra el resumen de los resultados de la encuesta de satisfacción completada por los alumnos de los cursos 2008 al 2011 de la asignatura Lógica Programable impartida en el tercer curso de Ingeniería en Electrónica Industrial. Dichos resultados han sido obtenidos por el profesor de la asignatura y en ella se utiliza el laboratorio remoto WebLab-CPLD.

Tabla 5.1. Encuestas de satisfacción en la asignatura Lógica Programable.

	Media 2008/ 2009	Media 2009/ 2010	Media 2010/ 2011	Media 2011/ 2012
Número de accesos	1.706	632	1.012	3.180
1. ¿Te ha ayudado el WebLab en la asignatura?	4,6	3,8	3,75	4,21
2. ¿Te parece una buena idea extenderlo a todos los alumnos de la Facultad?	4,7	4,2	4,1	4,21
3. ¿Cuántas maquetas crees que harían falta para los 95 alumnos?		3,7	3,3	2,97
4. ¿Es fácil de usar?	4,4	3,9	3,9	4,42
5. ¿Qué tal es la calidad de lo visto (la webcam)?	3,2	2,7	2,5	2,79
6. ¿Te has sentido cómodo con la gestión de entradas?	3,7	3,0	3,1	3,61
7. ¿Qué te parece el tiempo asignado para cada conexión?	3,7	3,1	2,4	3,67
8. ¿Qué te parecen las entradas/salidas seleccionadas?	3,8	3,4	3,5	3,97
9. Al estar alejado de la maqueta, ¿has tenido sensación de control?	4,1	3,6	3,7	3,85
10. ¿Te gustaría usar WebLab en otras asignaturas?	4,3	3,9	4,1	3,91
11. ¿Cuál es tu satisfacción global con el WebLab?	4,7	3,7	4	4,09
12. ¿Cuántas veces has tenido que esperar para usarlo?		2,1	2	3,94
13. ¿Cuántas veces lo has encontrado caído?		2,2	2,1	3,30
14 ¿Conoces a alguien que disfrute de un WebLab en otra universidad?	NO	NO	NO	NO

Más allá de un análisis detallado cabe destacar algunos aspectos singulares:

- La gestión de las entradas es considerada correcta por los alumnos aunque mejorable.

- El tiempo y los problemas de conexión (preguntas 7, 12 y 13) sugieren que la escalabilidad es la solución al acceso concurrente de múltiples usuarios. En este aspecto, la mejora de los resultados durante el último año encuestado se debe al despliegue de una segunda instancia de experimentación.
- La sensación de inmersión durante la experimentación es aceptable aunque la calidad de la webcam no sea muy apreciada por los alumnos (pregunta 5).

El anterior análisis no es parte fundamental del trabajo de investigación, pero clarifica que los resultados de la tesis doctoral no solo son válidos técnicamente sino que también resultan útiles y apreciables para los estudiantes.

5.5 Análisis comparativo entre laboratorios remotos para experimentación con tecnologías embebidas

Una vez validado el cumplimiento de las características identificadas como relevantes para el rendimiento de los laboratorios remotos que ofrecen experimentación con sistemas embebidos en las implementaciones llevadas a cabo para la evaluación de la arquitectura propuesta, se ha realizado una comparativa con los sistemas que fueron analizados en el estado del arte desarrollado durante el capítulo 2.

En la Tabla 5.2 se recoge el grado de cumplimiento de las características establecidas en el capítulo 2 por cada uno de los laboratorios implementados, así como por los laboratorios analizados en el estado del arte. Cada característica se representa en un color en base a tres posibles grados de cumplimiento:

- El color verde representa el cumplimiento total con la característica.
- El color amarillo representa el cumplimiento parcial con la característica.
- El color rojo representa el no cumplimiento con la característica.

Entre los 10 laboratorios remotos que fueron señalados como relevantes durante el estado del arte se puede observar como sólo 4 cumplen conjuntamente con las características de replicabilidad y escalabilidad:

- "ViciLab", laboratorio Remoto para experimentación con circuitos digitales y FPGA desarrollado por la Universidad Nacional de Irlanda en Galway (Morgan, y otros, 2014). Este laboratorio garantiza la escalabilidad soportando un alto número de instancias. Además permite cierta adecuación a nuevas tecnologías, pero únicamente se puede replicar sobre dispositivos programables puros comprometiendo la replicabilidad.
- Labshare FPGA Rig (Lowe, Murray, Lindsay, & Liu, 2009).La escalabilidad de este laboratorio remoto conlleva un alto coste económico puesto que cada

nueva instancia de experimentación exige la instalación de un nuevo computador que se encarga de realizar la grabación de la plataforma de experimentación. Por otro lado, la replicabilidad de este laboratorio remoto se limita a dispositivos FPGA de la serie Spartan que actualmente se encuentran discontinuados.

- Laboratorio de educación a distancia, MicroLab de la Universidad Demirel Sulayman (Kutlu & Taşdelen, 2010). Aunque la escalabilidad de este laboratorio permite el despliegue de un gran número de instancias, la replicabilidad se ve comprometido por la necesidad de implementar el servidor de grabación sobre un bootloader sobre puerto UART. Además el sistema que implementa las tareas del servidor de interacción únicamente soporta entradas digitales.
- REAL, Remote Engineering and Applications Laboratory de la Universidad Tecnológica de Illmenau (Henke, Ostendorff, Wuttke, & Vogel, 2012). Aunque este laboratorio fue descrito como referencia al permitir experimentar con diferentes tecnologías sobre diferentes dispositivos, únicamente soporta señales digitales, limitando la naturaleza de la experimentación. Además, la compartición de un bus entre sistemas de desarrollo y dispositivos controlables impide el acceso concurrente y dificulta el consumo desde dominios de estudiantes grandes. En este sentido la escalabilidad proporciona más ámbitos de experimentación, pero no acceso concurrente.

Es importante recordar que únicamente el cumplimiento de ambas características (escalabilidad y replicabilidad) permite garantizar la sostenibilidad del laboratorio remoto permitiendo adecuar las plataformas de experimentación a las nuevas tecnologías comercializadas por los fabricantes y adaptar el número de instancias ante un dominio variable de estudiantes.

La Tabla 5.2 muestra el cumplimiento de las características definidas como fundamentales (desplegabilidad, escalabilidad y replicabilidad), por los laboratorios implementados según la arquitectura abierta para el despliegue de laboratorios remotos para experimentación sobre tecnologías para el desarrollo de sistemas embebidos. Además puede observarse como la arquitectura no compromete el cumplimiento del resto de características generalistas para laboratorios remotos.

El cumplimiento parcial de la condición de replicabilidad en las implementaciones WebLab-PIC y WebLab-Robot se debe al empleo de un bootloader como servidor de grabación. Esto limita la replicabilidad hacia dispositivos basados en microprocesador.

Así mismo todas las implementaciones a excepción del laboratorio remoto WebLab-CPLD exigen la utilización de las herramientas proporcionadas por el

fabricante de la tecnología sobre la que basan la experimentación. Este acercamiento reduce sustancialmente los tiempos necesarios para el desarrollo de una sesión, pero compromete la universalidad del laboratorio puesto que los dispositivos desde los que se permite la experimentación heredan los requisitos que estas herramientas disponen para su instalación.

Tabla 5.2. Grado de cumplimiento de las características que definen el rendimiento de los laboratorios remotos para experimentación con tecnologías embebidas por los laboratorios implementados según la arquitectura propuesta en comparación con los laboratorios analizados en el estado del arte.

LR	*Replicabilidad*	Escalabilidad	Desplegabilidad	Integración	Disponibilidad	Conectividad	Universalidad
WebLab-PIC							
WebLab-Robot							
WebLab-Box							
WebLab-Elevator							
FPGA-FAU							
FPGA-UC							
ViciLab							
Labshare FPGA							
FPGA CUAS							
MCU UI							
MicroLab SDU							
Arduino HOU							
DSP UM							
REAL ITU							
FPGA HUST							
eDIVIDe							
FPGA JIANGSU							
FPGA UM							
FPGA SRMU							
FPGA UPC							
RexPIC IFCE							
PIC UR							
DistanceLab TUT							
PIC IIDTA							
FPGA UF							
MCU UP							
FPGA UNED							
PIC UNED							
Arduino Ukrania							

Capítulo

6

Conclusiones y líneas futuras

En este capítulo se presentan las conclusiones obtenidas como resultado del trabajo de investigación doctoral llevado a cabo, y se sugieren líneas futuras de investigación alineadas con los resultados y conclusiones.

En primer lugar, es momento de revisar la hipótesis inicial de la tesis doctoral según quedó enunciada en el capítulo 1.

H.	*Es posible diseñar e implementar a bajo coste una arquitectura que facilite el despliegue eficiente de nuevos laboratorios remotos que permitan la experimentación sobre tecnologías para el desarrollo de sistemas embebidos por cualquier persona, bajo un enfoque que garantice su sostenibilidad*

La hipótesis de partida ha quedado validada explícitamente en los capítulos 4 y 5, mientras que los capítulos 2 y 3 han servido para contextualizar el trabajo. Tras la justificación en el capítulo 1 de la importancia de los laboratorios remotos para la experimentación con tecnologías de desarrollo de sistemas embebidos, el capítulo 2 estableció el estado del arte, en el capítulo 3 se describe la nueva arquitectura en la que se basan las implementaciones descritas en el capítulo 4 y validadas en el capítulo 5.

El desarrollo del trabajo de investigación en los anteriores capítulos ha estado en consonancia con los objetivos específicos y operativos descritos en el capítulo 1. Dichos objetivos han sido cumplidos:

- El primer paso consistió en la búsqueda y discriminación de los principales laboratorios remotos para experimentación con sistemas embebidos desplegados por la comunidad investigadora. Del total de artículos revisados se seleccionaron 25 y de ellos se describieron y analizaron en detalle los 10 más significativos atendiendo a su arquitectura.

- Desde un punto de vista operativo y metodológico el anterior paso se fundamentó no solo en la lectura detallada de los artículos y memorias publicadas, sino especialmente en la propia experimentación sobre los laboratorios remotos y la medida del rendimiento utilizando herramientas informáticas.

- Como resultado del estado del arte, la tesis enmarcó el conjunto de laboratorios remotos analizados mediante un conjunto de características que fue clave para establecer los requisitos de la nueva arquitectura a diseñar. Esta caracterización es una aportación de la tesis en la medida que clarifica de una forma efectiva el estado del arte, lo que facilita la posterior evaluación.

- Una vez establecidos las limitaciones de las arquitecturas en las que se basan los laboratorios remotos del estado del arte, se establece una nueva arquitectura con el objetivo de cubrir de forma efectiva los requisitos de diseño ya establecidos. La nueva arquitectura planteada es genérica en su concepción y sobre todo supera de forma efectiva las limitaciones de las arquitecturas previas que dependían de su propia naturaleza. Una aportación de la tesis ha sido relacionar algunas de las principales características enunciadas en el estado del arte con aspectos relevantes de cada arquitectura. Es decir, cada arquitectura condiciona de forma significativa las distintas implementaciones basadas en ella y en ningún caso pueden superarlas pese a esfuerzos técnicos hardware y software.

- Para comprobar la validez de la arquitectura, el siguiente paso consistió en su despliegue. El capítulo 4 describe cuatro distintas implementaciones basadas en la nueva arquitectura, siendo cada una de ellas una prueba de concepto para cada una de las características definidas como cruciales en el capítulo 2. Dichas pruebas de concepto van desde una implementación ligera y completa de la nueva arquitectura, hasta una implementación profesional lista para su comercialización.

- Una vez implementadas las pruebas de concepto, el capítulo 5 las evalúa siguiendo la lista de requisitos y características descritas en el capítulo 2. Metodológicamente, dicha evaluación es cualitativa y cuantitativa. En primer lugar se relaciona el conjunto de requisitos con la propia arquitectura, siendo las pruebas de concepto las encargadas de fundamentar dicha evaluación. Desde el punto de vista cuantitativo, la tesis persigue no solo desplegar pruebas de concepto funcionales, sino también útiles, entendida "la utilidad" como el interés de la comunidad de usuarios (profesores y estudiantes) ante dichas implementaciones.

- El capítulo 5 finaliza comparando las características de los nuevos laboratorios implementados con las de los laboratorios remotos más significativos según el estado del arte. El resultado de la comparación destaca que claramente las nuevas implementaciones superan funcionalmente a las analizadas en el estado del arte.

El proceso descrito sigue las pautas de un trabajo de investigación doctoral cuyas las principales conclusiones son las siguientes:

- La desplegabilidad ha de ser tenida en cuenta como la principal característica para el consumo eficiente y sostenible de laboratorios remotos para experimentación con tecnologías de desarrollo de sistemas embebidos.

- Es posible diseñar una arquitectura genérica que asegure técnicamente su desplegabilidad en base a un nuevo conjunto de componentes: servidor de administración, servidor del laboratorio, cliente, servidor de interacción, servidor de grabación y plataforma de experimentación.

- Es posible implementar diversos laboratorios remotos sobre diferentes tecnologías (FPGA, CPLD, MCU, DSP, etc.) utilizando la arquitectura propuesta de manera que los sistemas diseñados sean fácilmente desplegables y adecuables a nuevos experimentos.

- Los nuevos laboratorios remotos implementados siguiendo la arquitectura diseñada son superiores funcionalmente a los analizados en el estado del arte.

Los resultados obtenidos de forma específica son los siguientes:

- Caracterización del rendimiento funcional de laboratorios remotos para la experimentación sobre tecnologías para el desarrollo de sistemas embebidos.

- La especificación de la "Arquitectura abierta para el despliegue de laboratorios remotos con experimentación sobre tecnologías de desarrollo de sistemas embebidos".

- Cuatro implementaciones de la nueva arquitectura en diferentes contextos experienciales: FPGA, MCU y CPLD con robots móviles, sistemas de desarrollo didácticos y maquetas industriales.

- Caracterización funcional de los laboratorios remotos implementados y estudio de los accesos por parte de los usuarios.

6.1 Publicaciones y resultados del trabajo de investigación

El desarrollo de la tesis doctoral ha permitido no solo la publicación de distintos trabajos científicos en revistas y congresos internacionales, sino también su explotación en proyectos nacionales e internacionales.

Los resultados de la investigación han permitido participar a la Facultad de Ingeniería de la Universidad de Deusto con el Instituto Tecnológico de Deusto, Deustotech en proyectos de investigación de carácter local, nacional e internacional. Destacan los proyectos internacionales:

- Proyecto "Building an ecology of Online Laboratories" (Award#: 1132813) subvencionado por la National Science Fundation bajo el programa Catalyzing New International cooperation. Años 2011-2012.
- Proyecto "ePragmatic: E-Learning and Practical Training of Mechatronics and Alternative Technologies in Industrial Community" (510586-LPP-1-2010-1-SI-LEONARDO-LNW) subvencionado por la Comunidad Europea bajo el programa Leonardo da Vinci, Thematic Network. Años 2010-2012.
- Proyecto "OLAREX: Open Learning Approach with Remote Experiments Project" (518987-LLP-1-2011-1-ES-KA3-KA3MP) subvencionado por la Comunidad Europea bajo el programa Lifelong Learning Program. Años 2011-2013.
- Proyecto "ICo-op: Industrial Cooperation and Creative Engineering Education based on Remote Engineering and Virtual Instrumentation" (530278-TEMPUS-1-2012-1-DE-TEMPUS-JPHES) subvencionado por la Comunidad Europea bajo el programa Tempus. Años 2012-2015.
- Proyecto "Nerela: Building Network of Remote Labs for strengthening university-secundary vocational schools collaboration" (543667-TEMPUS-1-2013-1-RS-TEMPUS-JPHES) subvencionado por la Comunidad Europea bajo el programa Tempus. Años 2013-2016.
- Proyecto "GO-LAB: Global Online Science Labs for Inquiry Learning at School" (Grant Agreement no. 317601) subvencionado por la Comunidad Europea bajo el 7º Programa Marco. Años 2012-2016.

El presupuesto total de los proyectos anteriores supera los 18 millones de euros y engloba a más de 50 socios de más de 20 países. Estos números dan idea de la utilidad de los resultados de investigación, más allá de su excelencia técnica.

Por otra parte, desde un punto de vista técnico e investigador, la tesis doctoral ha permitido la publicación y presentación de más de 25 trabajos en revistas y congresos. A continuación se citan los principales:

García-Zubia, J., Angulo, I., Hernández, U., & Orduña, P. (2008). Low Cost Remote Lab for Microcontrollers: WebLab-DEUSTO-PIC . Remote Engineering and Virtual Instrumentation (REV2008), International Conference in. Dusseldorf (Germany).

García-Zubia, J., Angulo, I., Hernández-Jayo, U., & Orduña, P. (2008). Plug&Play remote lab for microcontrollers: WebLab-DEUSTO-PIC. 7th workshop on microelectronics education in Europe (EWME 2008). Budapest.

García-Zubia, J., Angulo, I., Orduña, P., Irurzun, J., Ruiz de Garibay, J., Hernandez, U., & Gonzalez, J. (2009). Innovative Autonomous Hardware for Remote Experimentation with Microcontrollers. 3rd IEEE International Conference on e-Learning in Industrial Electronics - ICELIE'2009. Porto.

García-Zubia, J., Hernandez, U., Angulo, I., Orduña, P., & Irurzun, J. (2009). Acceptance, Usability and Usefulness of WebLab-Deusto from the Students Point of View. International Journal of Online Engineering – IJOE, 5(1), 26-31.

García-Zubia, J., Angulo, I., Hernandez, U., Castro, M., Sancristobal, E., Orduna, P., Ruiz de Garibay, J. (2010). Easily Integrable platform for the deployment of a Remote Laboratory for microcontrollers. International Journal of Online Engineering – IJOE, 6(3), 9-15.

Garcia-Zubia, J., Orduna, P., Angulo, I., Hernandez, U., Dziabenko, O., Lopez-Ipina, D., & Rodriguez-Gil, L. (2011). Application and user perceptions of using the WebLab-Deusto-PLD in technical education. *Frontiers in Education Conference (FIE)*, (págs. GOLC1.1-GOLC1.6). Rapid City. doi:10.1109/FIE.2011.6143127.

Campos, B., Angulo, I., Dziabenko, O., Orduna, P., Rodriguez, L., & Garcia-Zubia, J. (2012). Easily deployable low-cost remote lab platform. Remote Engineering and Virtual Instrumentation (REV), 2012 9th International Conference on. 1-4, Bilbao. doi: 10.1109/REV.2012.6293132

García-Zubia, J., Angulo, I., Orduña, P., Hernandez, U., Lopez-de-Ipina, D., Rodriguez, L., Dziabenko, O., Canivell, V. (2012). WebLab-Deusto-CPLD: A

Practical Experience. International Journal of Online Engineering – IJOE, 8(2012), 17-18.

Dziabenko, O., Rojko, A., Angulo, I., & Garcia-Zubia, J. (2012). Training of microcontrollers using remote experiments. 8th International Conference on Remote Engineering and Virtual Instrumentation. 1-6, Bilbao. doi: 10.1109/REV.2012.6293133.

Dziabenko, O., Orduña, P., García-Zubia, J. & Angulo, I. (2012). Remote Laboratory in Education: WebLab-Deusto Practice. In T. Bastiaens & G. Marks (Eds.), Proceedings of E-Learn: World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education 2012 (pp. 1445-1454). Chesapeake, VA: Association for the Advancement of Computing in Education (AACE).

Moreno, A. & Angulo, I. & Perallos, A. & Landaluce, H. & García-Zuazola, I. J. & Azpilicueta, L. & Astrain, J. J. & Falcone, F. & Villadongos, J. (2012). IVAN: Intelligent Van for the Distribution of Pharmaceutical Drugs. Sensors 2012, 12, 6587-6609; doi:10.3390/s120506587. Impact factor: 2.05.

Garcia-Zubia, J.; Angulo, I.; Dziabenko, O.; & Orduna, P. (2013). OLAREX project: Open learning approach with remote experiments. Global Engineering Education Conference (EDUCON), 2013 IEEE. 442-450, Berlin. doi: 10.1109/EduCon.2013.6530143

Iturrate, I., Angulo, I., & Orduña, P. (2013). Remote laboratory for serious games deployment based on a mobile robot platform. En O. Dziabenko, & J. García-Zubía (Edits.), *IT Innovative Practices in Secondary Schools: Remote Experiments* (págs. 393-320). Universidad de Deusto.

García-Zubía, J., Angulo, I., Rodriguez-Gil, L., Orduña, P., Dziabenko, O., & Güenaga, M. (2013). Boole-WebLab-FPGA: Creating an Integrated Digital Electronics Learning Workflow Through a Hybrid Laboratory and an Educational Electronics Design Tool. International Journal of Online Engineering – IJOE, 9, 19-22.

Orduña, P., Rodriguez-Gil, L., Angulo, I., Dziabenko, O., Hernandez-Jayo, U., López-De-Ipiña, D., & García-Zubía, J. (2014). Towards a microRLMS approach for shared development of remote laboratorios. 10th International Conference on Remote Engineering and Virtual Instrumentation. 375-381. Oporto. Doi:10.1109/REV.2014.6784192.

Angulo, I., & García-Zubia, J. (2014). Experimentación Remota sobre Maqueta Industrial Basada en un Ascensor de Tres Plantas. Revista Iberoamericana de Tecnologías del Aprendizaje - IEEE VAEP-RITA, 2(4), 199-204.

Angulo, I. & Onieva, E. & Perallos, A. & Salaberria, I. & Bahillo, A. & Azpilicueta, L. & Falcone, F. & Astrain, J. J. & Villadongos, J. (2015). Low Cost Real Time Location System Based in Radio Frequency Identification for the Provision of Social and Safety Services. Wireless Personal Communications, January 2015. doi: 10.1007/s11277-015-2767-6. Impact factor: 0.98

6.2 Líneas futuras

Una vez establecidos los resultados de este trabajo de investigación cabe señalar susceptibles futuras líneas de investigación:

- Extensión y validación de la nueva arquitectura a otras tecnologías embebidas y otros campos de experimentación para reafirmar su desplegabilidad.
- Adecuación de la arquitectura a los requisitos de accesibilidad de los usuarios con necesidades especiales.
- A nivel específico la arquitectura puede ser revisada para la inclusión de mecanismos de depuración, críticos en la evaluación de los sistemas embebidos.
- Determinación de una métrica cuantitativa para la validación funcional de los laboratorios remotos.
- Evaluación didáctica de los laboratorios remotos implementados, conectando analíticamente la arquitectura con los resultados obtenidos.
- Analizar los resultados tangibles de la tesis para evaluar la viabilidad de posibles patentes y/o modelos de utilidad.

Capítulo

6

Referencias

A7 engineering, i. (2009). *A7 EmbeddedBlue eb101 OEM Bluetooth Serial Adapter*. Recuperado el 10 de 07 de 2015, de http://www.coco3.com/community/wp-content/files/a7-ds-eb101.pdf

Allen, I., & Seamann, J. (2014). *Grade Change: Tracking Online Education in the United States*. Babson Survey Research Group. Recuperado el 10 de 07 de 2015, de http://www.onlinelearningsurvey.com/reports/gradechange.pdf

Altera. (2008). *Cyclone II Device Handbook, Volume 1*. Recuperado el 10 de 07 de 2015, de https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/hb/cyc2/cyc2_cii5v1.pdf

Altera. (2012). *DE2 User Manual*. Recuperado el 10 de 07 de 2015, de ftp://ftp.altera.com/up/pub/Altera_Material/12.1/Boards/DE2/DE2_User_Manual.pdf

Angulo, I., & García-Zubia, J. (2014). Experimentación Remota sobre Maqueta Industrial Basada en un Ascensor de Tres Plantas. *Revista Iberoamericana de Tecnologías del Aprendizaje - IEEE VAEP-RITA, 2*(4), 199-204.

Angulo, I., García Zubia, J., & Asunción, I. (2014). WebLab-Elevator: Laboratorio remoto para la experimentación con dispositivos lógicos programables . *Actas del XI Congreso en Tecnologías, Aprendizaje y Enseñanza de la Electrónica - TAEE 2014*, (págs. 254-259). Bilbao.

Anzhelika, P., Olga, G., Ivanov, E., Sokolyanskii, A., & Kurson, S. (2015). Development and application of remote laboratory for embedded systems design. *Remote Engineering and Virtual Instrumentation (REV), 2015 12th International Conference on*, (págs. 69-73). doi:10.1109/REV.2015.7087265

Araujo, A., & Cardoso, A. (2009). Pedagogical effectiveness of a remote lab for experimentation in Industrial Electronics. *E-Learning in Industrial Electronics, 2009. ICELIE '09. 3rd IEEE International Conference on*, (págs. 104-108). doi:10.1109/ICELIE.2009.5413201

Arduino. (2014). *Arduino Ethernet Rev3 WITHOUT PoE.* Recuperado el 10 de 07 de 2015, de http://www.farnell.com/datasheets/1658655.pdf

Arras, P., Henke, K., Tabunshchyk, G., & Van Merode, D. (2015). Iterative pattern for the embedding of remote laboratories in the educational process. *Remote Engineering and Virtual Instrumentation (REV), 2015 12th International Conference on*, (págs. 52-55). doi:10.1109/REV.2015.7087262

Atmark Techno, i. (2015). *JTAG-Blazer Product Info.* Recuperado el 10 de 4 de 2015, de http://suzaku-en.atmark-techno.com/series/jtag-blazer

Atmel, C. (2014). *ATmega48A/PA/88A/PA/168A/PA/328/P.* Recuperado el 12 de 07 de 2015, de ATMEL 8-BIT MICROCONTROLLER WITH 4/8/16/32KBYTES: http://www.atmel.com/images/atmel-8271-8-bit-avr-microcontroller-atmega48a-48pa-88a-88pa-168a-168pa-328-328p_datasheet_complete.pdf

Balamuralithara, B., & Woods, P. C. (2009). Virtual laboratories in engineering education: The simulation lab and remote lab. *Computer Applications in Engineering Education*, 108-118. doi:10.1002/cae.20186

Bochicchio, M., & Longo, A. (2010). Extending LMS with Collaborative Remote Lab Features. *Advanced Learning Technologies (ICALT), 2010 IEEE 10th International Conference on*, (págs. 310-314). doi:10.1109/ICALT.2010.89

Butime, J., Besiga, R., Bwonyo, A., Nakanwagi, V., Togboa, T., & Katumba, A. (2012). Design of online Digital Electronics laboratories based on the NI ELVIS II platform. *Remote Engineering and Virtual Instrumentation (REV), 2012 9th International Conference on*, (págs. 1-3). Bilbao. doi:10.1109/REV.2012.6293098

Capacity, R. E.-R. (2009). Remote Experiments With Mobile-Robot Hardware via Internet at Limited Link Capacity. *Industrial Electronics, IEEE Transactions on, 56*(12), 4798-4805. doi:10.1109/TIE.2009.2027898

Castro, M., Llamas, M., San Cristobal, E., Martín, S., Gil, R., Tawfik, M., . . . Vicent, L. (2010). Servicios para plataformas educativas: laboratorios y aplicaciones. *IX Congreso en tecnologías aplicadas a la enseñanza de la electrónica.* Madrid (Spain).

Copyright Custom Computer Services, I. (2015). *CCS C Compiler Manual.* Obtenido de http://www.ccsinfo.com/downloads/CReferenceManual.pdf

Corter, J. E., Esche, S. K., Chassapis, C., Ma, J., & Nickerson, J. V. (2011). Process and learning outcomes from remotely-operated, simulated, and hands-on student laboratories. *Computers and Education, 57*(3), 2054-2067.

Corter, J. E., Nickerson, J. V., Esche, S. K., & Chassapis, C. (2004). Remote versus hands-on labs: a comparative study. *Frontiers in Education, 2004. FIE 2004. 34th Annual*, *2*, págs. 17-21. doi:10.1109/FIE.2004.1408586

Corter, J. E., Nickerson, J. V., Esche, S. K., Chassapis, C., Im, S., & Ma, J. (Aug de 2007). Constructing reality: A study of remote, hands-on, and simulated laboratories. *ACM Transactions on Computer-Human Interaction (TOCHI), 14*(2). doi:10.1145/1275511.1275513

de Jong, T., Linn, M. C., & Zacharia, Z. C. (April de 2013). Physical and Virtual Laboratories in Science and Engineering Education. *Science, 340*, 305-308. doi:10.1126/science.1230579

del Alamo, J., Brooks, L., McLean, C., Hardison, J., Mishuris, G., Chang, V., & Hui , L. (2002). The MIT microelectronics WebLab: A Web-enabled remote laboratory for microelectronics device characterization. *2002 World Congr. Netw. Learn. Global Environ.*

Digilent. (2012). *Nexys 2 Reference Manual.* Recuperado el 10 de 07 de 2015, de http://digilentinc.com/Data/Products/NEXYS2/Nexys2_rm.pdf

Digilent, I. (2005). *Digilent Port Communications Programmers Reference Manual.* Recuperado el 10 de 4 de 2015, de www.digilentinc.com: Digilent Port Communications

Digilent, I. (2005). *Spartan-3 Starter Kit User Manual.* Recuperado el 21 de 3 de 2015, de https://www.digilentinc.com/Data/Products/S3BOARD/S3BOARD_RM.pdf

Digilent, I. (2010). *Digilent JTAG-USB Cable Reference Manual.* Recuperado el 13 de 4 de 2015, de http://digilentinc.com/Data/Products/JTAG-USB/JTAG-USB%20Cable_rm.pdf

Digilent, I. (2013). *Nexys 3 Reference Manual.* Recuperado el 13 de 05 de 2015, de http://digilentinc.com/Data/Products/NEXYS3/Nexys3_rm_V2.pdf

Dziabenko, O., Garcia-Zubia, J., & Angulo, I. (2012). Time to play with a microcontroller managed mobile bot. *Global Engineering Education Conference (EDUCON), 2012 IEEE*, (págs. 1-5). Marrakesh (Morocco). doi:10.1109/EDUCON.2012.6201097

El-Medani, W. M. (2008). FPGA Remote Laboratory for Hardware E-Leraning Courses. *Computational Technologies in Electrical and Electronics Engineering, 2008. SIBIRCON 2008. IEEE Region 8 International Conference on*, 106-109.

Feisel, L., & Rosa, A. (2005). The Role of the Laboratory in Undergraduate Engineering Education. *Journal of Engineering Education, 94*, 121-130.

Ferreira, J. M., & Cardoso, A. C. (2005). A Moodle extension to book online labs. *International Journal of Online Engineering, 1*(2), 1-4.

Ferreira, J., Nedić, Z., Machotka, J., Nafalski, A., & Göl, O. (2010). International collaborative learning using remote workbenches. *Annual Conference on Engineering and Technology Education*, (págs. 47-51). Pattaya (Thailand).

Fotopoulos, V., Anastasios, S. I., & Anastasios, F. (2013). Preparing a remote conducted course for microcontrollers based on Arduino. *7th International Conference in Open and Distance Learning (ICODL 2013).* Ellinogermaniki Agogi (Greece).

Froyd, J., Wankat, P., & Smith, K. (2012). Five Major Shifts in 100 Years of Engineering Education. *Proceedings of the IEEE, 100*(Special Centennial Issue), 1344-1360. doi:10.1109/JPROC.2012.2190167

Gabriel, C., & Ovidiu, P. (2003). Dynamic reconfiguration of system-on-chip devices. *Electronics Technology: Integrated Management of Electronic Materials Production, 2003. 26th International Spring Seminar on*, (págs. 154-157). doi:10.1109/ISSE.2003.1260505

García Zubía, J., López de Ipiña, D., Hernández Jayo, U., Orduña, P., & Trueba, I. (2006). Evolución del WebLab de la Universidad de Deusto. *VII Congreso de Tecnologías Aplicadas a la Enseñanza de la Electrónica.* Madrid (Spain).

García-Zubía, J., & Alves, G. R. (Edits.). (2011). *Using Remote Labs in Education: Two Little Ducks in Remote Experimentation.* Universidad de Deusto.

García-Zubía, J., Angulo, I., Dziabenko, O., & Orduña, P. (2012). "Lecciones del Proyecto ePragmatic de la UE. *X Congreso de Tecnologías Aplicadas a la Enseñanza de la Electrónica (TAEE 2012)*, (págs. 434-439). Vigo (Spain).

García-Zubia, J., Angulo, I., Hernández, U., & Orduña, P. (2008). Low Cost Remote Lab for Microcontrollers: WebLab-DEUSTO-PIC. *Remote Engineering and Virtual Instrumentation (REV2008), International Conference in.* Dusseldorf (Germany).

García-Zubia, J., Angulo, I., Hernandez, U., Castro, M., Sancristobal, E., Orduna, P., . . . de Garibay, J. (2010). Easily Integrable platform for the deployment of a Remote Laboratory for microcontrollers. *Education Engineering (EDUCON), 2010 IEEE*, (págs. 327-334). Madrid. doi:doi: 10.1109/EDUCON.2010.5492558

García-Zubia, J., Angulo, I., Hernández-Jayo, U., & Orduña, P. (2008). Plug&Play remote lab for microcontrollers: WebLab-DEUSTO-PIC. *7th workshop on microelectronics education in Europe (EWME 2008).* Budapest (Hungary).

García-Zubia, J., Angulo, I., Orduña, P., Irurzun, J., Ruiz de Garibay, J., Hernandez, U., & Gonzalez, J. (2009). Innovative Autonomous Hardware for Remote Experimentation with Microcontrollers. *3rd IEEE International Conference on e-Learning in Industrial Electronics - ICELIE'2009.* Porto (Portugal).

Garcia-Zubia, J., Lopez-de-Ipina, D., Orduna, P., Hernandez, U., Angulo, I., & Irurzun, J. (2008). Acceptance, usability and usefulness of WebLab-Deusto from students point

of view. *Digital Information Management, 2008. ICDIM 2008. Third International Conference on*, (págs. 899-904). doi:10.1109/ICDIM.2008.4746846

Garcia-Zubia, J., Orduna, P., Angulo, I., Hernandez, U., Dziabenko, O., Lopez-Ipina, D., & Rodriguez-Gil, L. (2011). Application and user perceptions of using the WebLab-Deusto-PLD in technical education. *Frontiers in Education Conference (FIE)*, (págs. GOLC1.1-GOLC1.6). Rapid City. doi:10.1109/FIE.2011.6143127

Garcia-Zubia, J., Orduna, P., Lopez-de-Ipina, D., & Alves, G. (Dec. de 2009). Addressing Software Impact in the Design of Remote Laboratories. *Industrial Electronics, IEEE Transactions on, 56*(12), 4757-4767. doi:10.1109/TIE.2009.2026368

García-Zubia, J., Orduña, P., Irurzun, J., Angulo, I., & Hernandez-Jayo, U. (2009). Integración del laboratorio remoto WebLab-Deusto en Moodle. *MoodleMoot.* Bilbao (Spain).

gateway4labs team. (2012). *gateway4labs 0.1 documentation*. Recuperado el 10 de 07 de 2015, de http://gateway4labs.readthedocs.org/en/latest/#

Gilibert, M., Picazo, J., Auer, M., Pester, A., & Cusidó, J. A. (2006). 80C537 Microcontroller Remote Lab for E-Learning Teaching. *iJOE International Journal of Online Engineering*.

Gomes, L., Patricio, G., Ferreira, R., & Costa, A. (2009). Remote experimentation for introductory digital logic course. *E-Learning in Industrial Electronics, 2009. ICELIE '09. 3rd IEEE International Conference on*, (págs. 98-103). doi:10.1109/ICELIE.2009.5413204

Goncalves de Moraes, A., & Mendes de Sales, A. K. (2012). Aplicacao de laboratorios remotos em microcontroladores pic. *XIX Congresso Brasileiro de Automática, CBA 2012.*, (págs. 3634-3641).

Grimheden, M., & Törngren, M. (2005). What is embedded systems and how should it be taught?-results from a didactic analysis. *ACM Transactions on Embedded Computing Systems (TECS), 4*(3), 633-651.

Hanke, K., Ostendorff, S., & Wuttke, H. D. (2012). A Concept for a Flexible and Scalable Infrastructure for Remote Laboratories. En D. Uckelmann, B. Scholz-Reiter, I. Rügge, B. Hong, & A. Rizzi (Edits.), *The Impact of Virtual, Remote, and Real Logistics Labs* (Vol. 282, págs. pp 13-24). Bremen, Germany: Springer Berlin Heidelberg. doi:10.1007/978-3-642-28816-6_2

Harward, V., del Alamo, J., Lerman, S., Bailey, P., Carpenter, J., DeLong, K., . . . Y. (2008). The iLab Shared Architecture: A Web Services Infrastructure to Build Communities of Internet Accessible Laboratories. *Proceedings of the IEEE*, 931-950. doi:10.1109/ JPROC.2008.921607

Hazarinov, M., & Perotto, D. (2012). *Ethernet bootloader for the ATmega328P / W5100.* Recuperado el 10 de 07 de 2015, de https://github.com/mharizanov/TFTP Bootloader _0_2.

Henke, K., Ostendorff, S., Wuttke, H., & Vogel, S. (2012). A grid concept for reliable, flexible and robust remote engineering laboratories. *Remote Engineering and Virtual Instrumentation (REV), 2012 9th International Conference on*, (págs. 1-8). doi:10.1109/REV.2012.6293110

Henke, K., Tabunshchyk, G., Wuttke, H.-D., Vietzke, T., & Ostendorff, S. (2014). Using Interactive Hybrid Online Labs for rapid prototyping of digital systems. *Remote Engineering and Virtual Instrumentation (REV), 2014 11th International Conference on*, (págs. 61-66). doi:10.1109/REV.2014.6784222

Henke, K., Vietzke, T., Wuttke, H. D., & Ostendorff, S. (2015). Safety in Interactive Hybrid Online Labs. *International Journal of Online engineering, 11*(3), 56-61. doi:10.3991/ijoe.v11i3.4557

Herbert Yeung, D. L. (October de 2010). Interoperability of Remote Laboratories Systems. *International Journal of Online Engineering (iJOE), 6*, 71-80. doi:10.3991/ijoe.v6s1.1387

Hercog, D., & Jezernik, K. (2005). Rapid Control Prototyping using MATLAB/Simulink and DSP-based Motor Controller. *International Journal of Engineering Education (IJEE), 21*(4).

Hercog, D., Gergic, B., Uran, S., & Jezernik, K. (Dec de 2007). A DSP-Based Remote Control Laboratory. *Industrial Electronics, IEEE Transactions on, 54*(6), 3057-3068. doi:10.1109/TIE.2007.907009

Hernández-Jayo, U. (2012). Metodología de Control Independiente de instrumentos y Experimentos para su Despliegue en Laboratorios Remotos. *Tesis doctoral*. Bilbao: Universidad de Deusto.

HI-TECH, S. (8 de 2003). *PIC-C18 ANSI C Compiler.* Recuperado el 11 de 6 de 2015, de http://www.htsoft.com/: http://www.ee.bgu.ac.il/~microlab/MicroLab/PicDoc/manual.pdf

Hoang, T., Quang, H., Hung, T., de Souza-Daw, T., Ngoc, L., Dzung, N., & Bien, P. (2015). A Low-cost Remote Laboratory of Field Programmable Gate Arrays. *12th International Conference on Remote Engineering and Virtual Instrumentation, At Bangkok*, (págs. 172-176). Bangkok (Thailand).

Hui, Z., & Tie-jun, X. (2008). An Innovative Remote Experiment System for FPGA-Based Curriculum. *Proceedings of 2008 IEEE International Symposium on IT in Medicine and Education*, (págs. 870-875).

IEEE Computer Society. (2013). *Boundary Scan Architecture - Standard Test Access and Boundary Scan Architecture WG P1149.1.* Obtenido de https://standards.ieee.org/findstds/standard/1149.1-2013.html

INE, I. N. (2014). *Encuesta sobre Equipamiento y Uso de Tecnologías de Información y Comunicación en los Hogares.* Notas de Prensa, Instituto Nacional de Estadística (INE).

Ingeniería de Microsistemas Programados, S. (Oct. de 2008). *Manual de usuario de PIC'Control y PIC'Project.* Obtenido de www.microcontroladores.com.

Ingeniería de Microsistemas Programados, S. (Jul de 2010). *Azkar-Bot Manual de Usuario.* Obtenido de www.microcontroladores.com.

Ingeniería de Microsistemas, S. (2010). *Laboratorio USB-PIC'School Manual de usuario.* Obtenido de www.microcontroladores.com.

Iturrate, I., Angulo, I., & Orduña, P. (2013). Remote laboratory for serious games deployment based on a mobile robot platform. En O. Dziabenko, & J. García-Zubía (Edits.), *IT Innovative Practices in Secondary Schools: Remote Experiments* (págs. 393-320). Universidad de Deusto.

Jezernik, D. H. (2005). Rapid Control Prototyping using MATLAB/Simulink and a DSP-based Motor Controller. *Internationa Journal of Engineering Education, 21*(4), 596-605.

Johnson, L., Adams Becker, S., Estrada, V., & Martín, S. (2013). *Technology Outlook for STEM+ Education 2013-2018: An NMC Horizon Project Sector Analysis.* Austin, Texas: The New Media Consortium.

Johnson, M., & Miller, H. (2013). *Installing Moodle.* Recuperado el 12 de 07 de 2015, de Moodle docs: https://docs.moodle.org/23/en/Talk:Installing_Moodle

Joshi, P. V., & Gurumurthy, K. S. (2014). Analysing and improving the performance of software code for Real Time Embedded systems. *Devices, Circuits and Systems (ICDCS), 2014 2nd International Conference on*, (págs. 1-5). doi:10.1109/ICDCSyst.2014.6926134

Jung, S. (2013). Experiences in Developing an Experimental Robotics Course Program for Undergraduate Education. *Education, IEEE Transactions on, 56*(1), 129-136. doi:10.1109/TE.2012.2213601

Karthik, S., Shreya2, P., & Srihari, P. a. (2014). Remote Field-Programmable Gate Array (FPGA) Lab. *IJRET: International Journal of Research in Engineering and Technology, 03*(04), 842-845. doi:10.15623/ijret.2014.0304149

Kępa, K., Morgan, F., Kościuszkiewicz, K., Braun, L., Hübner, M., & Becker, J. (Dec. de 2010). Design Assurance Strategy and Toolset for Partially Reconfigurable FPGA Systems. *ACM Trans. Reconfigurable Technol. Syst., 4*(1). doi:10.1145/1857927.1857931

Krushinitskiy, P., & Sziebig, G. (2013). Review of open source computing devices for iSpace in production workshops. *Cognitive Infocommunications (CogInfoCom), 2013 IEEE 4th International Conference on*, (págs. 677-682). doi:10.1109/CogInfoCom.2013.6719187

Kumar, A., Fernando, S., & Panicker, R. (Nov de 2013). Project-Based Learning in Embedded Systems Education Using an FPGA Platform. *IEEE Transactions on Education, 56*(4), 407-415. doi:10.1109/TE.2013.2246568

Kutlu, A. (2004). MicroLab: A Web-based Multi-user. *International Journal of Engineering Education, 20*(5), 879-885.

Kutlu, A., & Taşdelen, K. (2010). Remote electronic experiments using LabVIEW over controller area network. *Scientific research and essays, 5*, 1754-1758.

Lacey, A., & Wright, B. (2009). *Occupational employment projections.* Monthly Labour Report.

Limpraptono, F. Y., Ratna, A. A., & Sudibyo, H. (2012). Remote laboratories multiuser based on embedded web server. *Remote laboratories multiuser based on embedded web server*, (págs. 1-7). doi:10.1109/REV.2012.6293113

Limpraptono, F., Sudibyo, H., Ratna, A., & Arifin, A. (2011). The design of embedded web server for remote laboratories microcontroller system experiment. *TENCON 2011 - 2011 IEEE Region 10 Conference*, (págs. 1198-1202). doi:10.1109/TENCON.2011.6129302

Lindsay, E., Liu, D., Murray, S., & Lowe, D. (2007). Remote laboratories in Engineering Education: Trends in Students Perceptions. *Eighteenth Annual Conference of the Australasian Association for Engineering Education.*

Lobo, J. (2011). Interactive Demonstration of a Remote Reconfigurable Logic Laboratory for Basic Digital Design. *Proceedings of the 1st Experiment@ International Conference - Remote & Virtual Labs - exp.at11.* Lisbon (Portugal).

Lowe, D., Machet, T., & Kostulski, T. (2012). UTS Remote Labs. Labshare, and the Sahara Architecture. En *Using Remote Labs in Education: Two Little Ducks in Remote Experimentation* (págs. 403-424). Universidad de Deusto.

Lowe, D., Murray, S., Lindsay, E., & Liu, D. (oct de 2009). Evolving Remote Laboratory Architectures to Leverage Emerging Internet Technologies. *Learning Technologies, IEEE Transactions on, 2*(4), 289-294. doi:10.1109/TLT.2009.33

Ma, J. a. (2006). Hands-on, Simulated and Remote Laboratories: A Comparative Literature Review. *ACM Computing Surveys, 38*(3), 1-24.

Microchip Technolgy, I. (2009). *PIC18F2455/2550/4455/4550 Data Sheet.* Recuperado el 11 de May. de 2015, de http://ww1.microchip.com/downloads/en/DeviceDoc/39632e.pdf

Microchip Technology, I. (2002). *A FLASH Bootloader for PIC16 and PIC18 Devices.* Recuperado el 13 de May de 2015, de http://ww1.microchip.com/downloads/en/AppNotes/00851b.pdf

Microchip Technology, I. (2006). *PICDEM.net 2 Internet/Ethernet Development Board.* Recuperado el 23 de Mayo de 2015, de http://ww1.microchip.com/downloads/en/DeviceDoc/51623a.pdf

Microchip Technology, I. (2007). *8-Bit I/O Expander with Serial Interface.* Recuperado el 09 de 07 de 2015, de http://ww1.microchip.com/downloads/en/DeviceDoc/21919e.pdf

Microchip Technology, I. (2008). *The Microchip TCP/IP Stack.* Recuperado el 17 de May de 2015, de http://ww1.microchip.com/downloads/en/AppNotes/00833c.pdf

Microchip Technology, I. (2010). *MCP4902/4912/4922 8/10/12-Bit Dual Voltage Output Digital-to-Analog Converter.* Recuperado el 10 de 4 de 2015, de http://ww1.microchip.com/downloads/en/DeviceDoc/22250A.pdf

Microchip Technology, I. (24 de 11 de 2010). *Microchip COFF Help.* Obtenido de www.microchip.com.

Microchip Technology, I. (2010). *PICKit 3 User Guide.* Recuperado el 10 de 4 de 2015, de http://ww1.microchip.com/downloads/en/DeviceDoc/51795B.pdf

Microchip Technology, I. (2011). *Low-Cost I2 Real-Time Clock/Calendar wih SRAM and Battery Switchover.* Recuperado el 09 de 07 de 2015, de http://ww1.microchip.com/downloads/en/DeviceDoc/25010A.pdf

Microchip Technology, I. (2 de agosto de 2011). *PIC18F97J60 Family Data Sheet.* Recuperado el 17 de Mayo de 2015, de http://ww1.microchip.com/downloads/en/DeviceDoc/39762f.pdf

Microchip Technology, I. (2012). *PIC18(L)F2X/4XK22 Data Sheet.* Recuperado el 12 de May de 2015, de http://ww1.microchip.com/downloads/en/DeviceDoc/41412F.pdf

Microchip Technology, I. (2013). *256K I2C™ CMOS Serial EEPROM.* Recuperado el 09 de 07 de 2015, de http://ww1.microchip.com/downloads/en/DeviceDoc/20001203U.pdf

Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., . . . Martinoli, A. (2009). The e-puck, a Robot Designed for Education in Engineering. *Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions*, (págs. 59-65). Castelo Branco (Portugal).

Morgan, F., & Cawley, S. (2011). Enhancing learning of digital systems using a remote FPGA lab. *Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), 2011 6th International Workshop on*, (págs. 1-8). doi:10.1109/ReCoSoC.2011.5981525

Morgan, F., Cawley, S., & Newell, D. (October de 2012). Remote FPGA Lab for Enhancing Learning of Digital Systems. *ACM Trans. Reconfigurable Technol. Syst., 5*(3), 18:1--18:13. doi:10.1145/2362374.2362382

Morgan, F., Cawley, S., Callaly, F., Agnew, S., Rocke, P., O'Halloran, M., . . . Mc Ginley, B. (2011). Remote FPGA Lab with interactive control and visualisation interface. *21st International Conference on Field Programmable Logic and Applications (FPL 2011).*

Morgan, F., Cawley, S., Coffey, A., Callaly, F., Lyons, D., O'Loughlin, D., . . . Killoran, P. (2014). viciLogic: Online learning and prototyping platform for digital logic and computer architecture. *eChallenges e-2014, 2014 Conference*, (págs. 1-9).

Mougharbel, I., El Hajj, A., Artail, H., & Riman, C. (2006). Remote Lab Experiments Models: A Comparative Study. *Internationa Journal of Engineering, 22*(4), 49±857.

Murray, S., Lowe, D., Lindsay, E., Lasky, V., & Liu, D. (2008). Experiences with a Hybrid Architecture for Remote Laboratories. *The 38th Annual Frontiers in Education Conference.* Saratoga Springs.

Nedic, Z., Machotka, J., & Nafalski, A. (2003). Remote laboratories versus virtual and real laboratories. *Frontiers in Education, 2003. FIE 2003 33rd Annual, 1*, págs. 5-8. doi:10.1109/FIE.2003.1263343

Nickerson, J. V., Corter, J. E., Eschea, S. K., & Chassapis, C. (November de 2007). A model for evaluating the effectiveness of remote engineering laboratories and simulations in education. *Computers & Education, 49*(3), 708-725. doi:10.1016/j.compedu.2005.11.019

Orduna, P., Rodriguez-Gil, L., & Lopez-de-Ipina, D. a.-Z. (2012). Sharing the remote laboratories among different institutions: A practical case. *Remote Engineering and Virtual Instrumentation (REV), 2012 9th International Conference on*, (págs. 1-4). doi:10.1109/REV.2012.6293178

Orduña, P. (2013). Transitive and Scalable Federation Model for Remote Laboratories. *Tesis doctoral*. Bilbao: Universidad de Deusto.

Orduña, P., Botero, S., Hock, N., Sancristobal, E., Emaldi, M., Pesquera, A., . . . García-Zubia, J. (2013). Generic integration of remote laboratories in learning and content management systems through federation protocols. *2013 IEEE Frontiers in Education Conference*, (págs. 1372-1378). Oklahoma City, OK, USA. doi:10.1109/FIE. 2013.6685057

Orduña, P., Caminero, A., Lequerica, I., Zutin, D., Bailey, P., Sancristobal, E., . . . García-Zubía, J. (2014). Generic integration of remote laboratories in public learning tools: organizational and technical challenges. *2014 IEEE Frontiers in Education Conference.*

Orduña, P., García-Zubia, J., Irurzun, J., & Rodriguez-Gil, L. (2011). Enabling mobile access to Remote Laboratories. *Global Engineering Education Conference (EDUCON), 2011 IEEE*, (págs. 312-318). doi:10.1109/EDUCON.2011.5773154

Orduña, P., García-Zubia, J., López-de-Ipiña, D., Bailey, P., Hardison, J., DeLong, K., & Harward, J. (2012). Achieving interoperability among educational remote laboratories. Rome (Italy).

Orduña, P., Irurzun, J., Rodriguez-Gil, L., García-Zubia, J., Gazzola, F., & López-de-Ipiña, D. (2011). Adding New Features to New and Existing Remote Experiments through their

Integration in WebLab-Deusto. *International Journal in Online Education (IJOE)*, 33-39. doi:10.3991/ijoe.v7iS2.1774

Pop, D., Zutin, D. G., Auer, M. E., Henke, K., & Wuttke, H. D. (2011). An Online Lab to Support a Master Program in Remote Engineering. *Frontiers in Education Conference (FIE), 2011*, (págs. GOLC2-1-GOLC2-6). doi:10.1109/FIE.2011.6143131

Powell, A. (2012). Democratizing Production Through Open-Source Knowledge: From open software to open hardware. *Media, Culture and Society, 34*(6), 691-708.

Rangel, C. V., Chacón, R. A., & Araque, A. A. (2011). Desarrollo de un laboratorio para microcontroladores con opción de acceso remoto. *Revista Colombiana de Tecnologías Avanzadas*, 136-139.

Reichenbach, M., Schmidt, M., Pfundt, B., & Fey, D. (2011). A New Virtual Hardware Laboratory for Remote FPGA Experiments on Real Hardware. *Proceedings of the 2011 International Conference on E-Learning, E-Business, Enterprise Information Systems, &E-Goverment (EEE 2011)* (págs. 17-23). Las Vegas: Vol.1, 1. CSREA Press.

Reimers, S., & Stewart, N. (2015). Presentation and response timing accuracy in Adobe Flash and HTML5/JavaScript Web experiments. *Behavior Research Methods, 47*(2), 309-327. doi:10.3758/s13428-014-0471-1

Restivo, M. T., & Cardoso, A. (2012). The Portuguese Contribution for lab2go - pt.lab2go. *International Journal of Online Engineering*.

Restivo, M., Mendes, J., Lopes, A., Silva, C., & Chouzal, F. (Dec de 2009). A Remote Laboratory in Engineering Measurement. *Industrial Electronics, IEEE Transactions on, 56*(12), 4836-4843. doi:10.1109/TIE.2008.2011479

Richter, T., Tetour, Y., & Boehringer, D. (2011). Library of Labs-A European Project on the Dissemination of Remote Experiments and Virtual Laboratories. *IEEE International Symposium on Multimedia (ISM2011)*, (págs. 543-548).

Rochadel, W., da Silva, S., da Silva, J., Luz, T., & Alves, G. (2012). Utilization of remote experimentation in mobile devices for education. *Global Engineering Education Conference (EDUCON), 2012 IEEE*, (págs. 17-20). Marrakesh. doi:10.1109/EDUCON.2012.6201112

Rojko, A., Hercog, D., & Jezernik, K. (Oct. de 2010). Power Engineering and Motion Control Web Laboratory: Design, Implementation, and Evaluation of Mechatronics Course. *Industrial Electronics, IEEE Transactions on, 57*(10), 3343-3354. doi:10.1109/TIE.2009.2031189

Rojko, A., Jezernik, K., & Pester, A. (2011). International E-PRAGMATIC network for adult engineering education. *Global Engineering Education Conference (EDUCON), 2011 IEEE*, (págs. 34-39). Amman (Jordan). doi:10.1109/EDUCON.2011.5773109

Saini, D. K. (2012). Software Testing for Embedded Systems. *International Journal of Computer Applications (0975-8887)*, Volume 43 - N0. 17.

San Cristóbal, E. (2010). Metodología, estructura y desarrollo de interfaces intermedias para la conexión de laboratorios remotos y virtuales a plataformas educativas. *Tesis doctoral*. Escuela Técnica Superior de Ingenieros Industriales Universidad Nacional de Educación a Distancia.

San Cristobal, E., Castro, M., Harward, J., Baley, P., DeLong, K., & Hardison, J. (2011). Integration view of Web Labs and Learning Management Systems. *Education Engineering (EDUCON), 2010 IEEE*, (págs. 1409-1417). doi:10.1109/EDUCON. 2010.5492363

San Cristobal, E., Pesquera, A., Orduña, P., Ruiz, E., Gil, R., Martin, S., Diaz, G., Albert, M. J., Colmenar, A., Meier, R., & Castro, M. (2014). Virtual and Remote Industrial Laboratory. Integration in Learning Management Systems. *IEEE Industrial Electronics Magazine*, 8(4) (págs. 45-58). doi: 10.1109/MIE.2012.2235530

Sánchez-Román, D., Sutter, G., López-Buedo, S., González, I., Gómez-Arribas, F., & Aracil, J. (2010). Virtualización de coprocesadores reconfigurables en sistemas HPRC con arquitectura Multicore. *Jornadas de Computación Reconfigurable y Aplicaciones (JCRA10).* Valencia.

Sell, R., Ruutmann, T., Murtazin, K., Kori, K., Pedaste, M., & Altin, H. (2015). Online Tools and Remote Labs for Making ICT More Atractive for Students to Prevent Dropout. *2015 IEEE Global Engineering Education Conference (EDUCON)* (págs. 804-808). Tallin (Estonia): IEEE.

Soares, J., & Lobo, J. (2011). A Remote FPGA Laboratory for Digital Design Students. *Proceedings of the 7th Portuguese Meeting on Reconfigurable Systems - REC2011*, (págs. 95-98). Porto (Portugal).

Tawfik, M., Sancristobal, E., Martin, S., Diaz, G., & Castro, M. (2012). State-of-the-art remote laboratories for industrial electronics applications. *Technologies Applied to Electronics Teaching (TAEE), 2012*, (págs. 359-364). Vigo (Spain).

Tawfik, M., Sancristobal, E., Martin, S., Diaz, G., Peire, J., & Castro, M. (2012). Expanding the Boundaries of the Classroom. Implementation of Remote Laboratories for Industrial Electronics Disciplines. *IEEE Industrial Electronics Magazine*, 7(1) (págs. 41-49). Vigo (Spain).

Terkowsky, C., Haertel, T., Bielski, E., & May, D. (2013). Creativity@ School: mobile learning environments involving remote labs and E-Portfolios. A conceptual framework to foster the inquiring mind in secondary STEM education. En *IT Innovative Practices in Secondary Schools: Remote Experiments.* (págs. 255-280). Bilbao: Universidad de Deusto.

Transparency Market Research. (2013). *Embedded System Market - Global Industry Analysis, Size, Share, Growth, Trends and Forecast 2012 - 2018.* Obtenido de http:// www.transparencymarketresearch.com/embedded-system.html

TU-Ilmenau.de/ics. (2013). *ICo-op: Industrial Cooperation and Creative Engineering Education based on Remote Engineering and Virtual Instrumentation.* Recuperado el 21 de 06 de 2015, de http://www.ico-op.eu/

UBM Tech. (2014). *2014 Embedded Market Study.* Recuperado el 23 de 06 de 2015, de http://bd.eduweb.hhs.nl/es/2014-embedded-market-study-then-now-whats-next.pdf

Uran, S., Hercog, D., & Jezernik, K. (2006). MATLAB Web Server and Web-based Control Design Learning. *IEEE Industrial Electronics, IECON 2006 - 32nd Annual Conference on*, (págs. 5420-5425). doi:10.1109/IECON.2006.347924

Vandorpe, J., Vliegen, J., Smeets, R., Mentens, N., Drutarovsky, M., Varchola, M., . . . Torresen, J. (2013). eDiViDe: European Digital Virtual Design Lab. *Annual Conference of the Europen Society for Engineering Education, SEFI 2013.* Leuven (Belgium).

Vassilis Fotopoulos, S. I. (2013). Preparing a remote conducted course for microcontrollers based on Arduino. *7th International Conference in Open and Distance Learning (ICODL 2013).* Ellinogermaniki Agogi, Athens (Greece).

Villar, E. (2001). *Design of Hardware/software Embedded Systems.* Universidad de Cantabria.

Waldrop, M. M. (2013). Education online: The virtual lab. Confronted with the explosive popularity of online learning researchers are seeking new ways to teach the practical skills of science. *Nature, 495*, 268-270.

Waterson, S., Landay, J., & Matthews, T. (2002). In the Lab and out in the Wild: Remote Web Usability Testing for Mobile Devices. *Extended Abstracts on Human Factors in Computing Systems* (págs. 796-797). ACM. doi:10.1145/506443.506602

Watterson, P., Yeoh, L., Giampietro, C., Chapman, C., & Houghton, L. (2010). Eddy current damper for the Labshare remote laboratory Shake Table rig. *Universities Power Engineering Conference (AUPEC), 2010 20th Australasian*, (págs. 1-6). Christchurch (New Zealand).

WebLab-Deusto, A. (2014). *Installation of the WebLab-Deusto RLMS.* Recuperado el 12 de 07 de 2015, de WebLab-Deusto 5.0 Documentation: http://weblabdeusto.readthedocs.org/en/latest/installation.html

Wolf, W., & Madsen, J. (2000). Embedded systems education for the future. *Proceedings of the IEEE, 88*(1), 23-30. doi:10.1109/5.811598

Xilinx. (2009). *Spartan-3E FPGA Family Data Sheet.* Recuperado el 11 de May de 2015, de http://www.xilinx.com/support/ documentation/data_sheets/ds312.pdf

Xilinx. (2013). *XC9572 In-System Programmable CPLD.* Recuperado el 11 de May de 2015, de http://www.xilinx.com/support/documentation/data_sheets/ds065.pdf

Xilinx. (2015). *Spartan-6 FPGA Data Sheet: DC and Switching Characteristics.* Recuperado el 11 de May de 2015, de http://www.xilinx.com/support/documentation/data_sheets/ds162.pdf

Yen, T.-Y., & Wolf, W. (1996). Hardware-Software Co-Synthesis of Distributed Embedded Systems. En T.-Y. Yen, & W. Wolf, *Hardware-Software Co-Synthesis of Distributed Embedded Systems.* New Tork: Springer Science Media.

Zenzerović, P., & Sučić, V. (2011). Remote Laboratory for. *MIPRO, 2011 Proceedings of the 34th International Convention*, (págs. 1685-1688). Opatija (Croacia) .

Zutin, D., Andrade, T. F., Restivo, M. T., & Rodrigues Quintas, M. (2013). Using ISA services to manage lab sessions with embedded lab servers. *Experiment@ International Conference (exp.at'13)*, (págs. 133-137). doi:10.1109/ExpAt.2013.6703045

Zutin, D., Auer, M., Maier, C., & Niederstatter, M. (2010). Lab2go — A repository to locate educational online laboratories. *Education Engineering (EDUCON)* (págs. 1741-1746). IEEE. doi:10.1109/EDUCON.2010.5492412