UNIVERSIDAD DE VALLADOLID

EDIFICIO DE LAS TECNOLOGÍAS DE LA

INFORMACIÓN Y LAS COMUNICACIONES

TRABAJO FIN DE MÁSTER

MÁSTER UNIVERSITARIO DE INVESTIGACIÓN

EN TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES

Diseño de un sistema de riego agrícola controlado a distancia

Autor:

D. Jesús Monge Álvarez

Tutora:

Dra. Dª. Lourdes Pelaz

Valladolid, 10 de Septiembre de 2015

TÍTULO:	**Diseño de un sistema de riego agrícola controlado a distancia**
AUTOR:	**D. Jesús Monge Álvarez**
TUTOR:	**Dra. Dª. Lourdes Pelaz**
DEPARTAMENTO:	

TRIBUNAL	
PRESIDENTE:	**Dr. D.**
VOCAL:	**Dr. D.**
SECRETARIO	**Dr. D.**
FECHA:	**Pendiente**
CALIFICACIÓN:	

Resumen de TFM

En este TFM se plantea el diseño de un prototipo que constituya una innovación incremental de una de las tecnologías de riego agrícola más recientes: los pívots de regadío. Los dos objetivos que se pretenden conseguir con este prototipo son: un uso más eficiente del agua empleada en el regadío agrícola y evitar largos desplazamientos para la simple puesta en marcha o monitorización del pívot.

Su parte hardware está basada en la placa computadora Raspberry Pi junto con la placa Custard Pi 2 para la interconexión con otros dispositivos (sensores, actuadores, etc.) presentes en el pívot. Estos permiten la monitorización de su estado y la detección de eventos de riesgo (presión de abastecimiento de agua fuera de los límites adecuados, apagado espontáneo, etc.). Su parte software emplea el lenguaje de programación PHP para ofrecer al usuario final una interfaz web que le permita controlar y monitorizar el pívot remotamente. Para la lectura de los dispositivos externos conectados se emplea el lenguaje de programación Python.

Los principales criterios considerados durante el diseño son: (1) que la interfaz web tenga un grado de usabilidad adecuado tanto para dispositivos fijos (PC) como móviles (smartphones, tabletas, etc.) – lo que se logró mediante la tecnología jQuery Mobile; (2) que el prototipo disponga de un sistema de alertas que avise al usuario final siempre que suceda un evento de riesgo – se implementó mediante el envío automático de correos electrónicos; (3) que el prototipo en sí mismo sea una tecnología eficiente, a fin de reducir al máximo posible sus costes de implementación y operación – lo que aumentará sus posibilidades de transferencia tecnológica al

área comercial. Una primera implementación del prototipo permitió verificar que se cumplían estos tres criterios de diseño.

Para conseguir que el prototipo permitiera evitar largos desplazamientos a los agricultores se debía lograr un control remoto del mismo, por lo que había que dotarlo de algún tipo de conectividad. Para ello, considerando el tipo de entorno donde se espera que el prototipo entre en funcionamiento, se optó por usar la red telefonía móvil (2G, 3G y 4G). Dado que a priori las tarifas más asequibles y flexibles son las ofertadas por las operadoras móviles virtuales, se hizo un análisis comparativo de la oferta de servicios de algunas de ellas.

Nuestro prototipo constituye un pequeño intento de inclusión de las Tecnologías de la Información y las Comunicaciones (TICs) en el mundo agrícola. Gracias a las TICs, se podrá lograr una gestión de las explotaciones agrarias basadas en principios científicos, lo que beneficiará tanto al agricultor – su explotación será más efectiva y eficiente – como a la sociedad en general – pues permiten reducir la huella medioambiental derivada de la actividad humana en ese sector productivo.

Palabras clave

Agua, agricultura, eficiencia, pívots de regadío, TICs.

Abstract

This work proposes a design for a prototype which is intended to become an incremental innovation for one the most recent agricultural technologies: the irrigation pivots. The prototype is aimed to achieve the following goals: a more efficient use of the water employed in the irrigation as well as avoiding long journeys simply to turn on/off the pivot or to control its performance.

The hardware part is based on the single-board computer Raspberry Pi together with the Custard Pi 2 board for connecting the prototype to some elements of the pivot such as sensors, actuator, etc. These elements allow a monitoring of the pivot and they can be used to detect risk events (loss of water supply pressure, unexpected shutdown, etc.). The software part employs PHP programming language in order to offer the final user a web interface that enables them a remotely controlling and monitoring of the pivot. For reading the information which comes from the external elements of the pivot that are connected to the prototype, Python language programming is used.

The major criteria considered during the design process are: (1) the interface web must have a proper degree of usability for both PCs and mobile devices (smartphones, tablets, laptops, etc.) – this is achieved by means of jQuery Mobile technology; (2) the prototype must offer the final user a system of alerts to warn them whenever a risk event takes place – automatic emailing is the basis of the alert system; (3) the prototype must constitute itself as an efficient technology, so as to

increase the probabilities of becoming a technological transfer to the commercial market. A simple implementation of the prototype allowed us to verify that the design fulfilled these three criteria.

A remotely usage of the prototype by the final user requires that it has some kind of connectivity. Considering the environment where the prototype is expected to be used, 2G, 3G, and 4G mobile networks were chosen for this purpose. Taking into account the most affordable and flexible tariffs are usually provided by mobile virtual network operators, we carried out a comparative analysis of the services of some of them.

Our prototype is a first attempt at inclusion of the Information and Communication Technologies (ICTs) in the agricultural world. Thanks to ICTs, we may achieve a management of the agricultural farms based on scientific principles, which will benefit both the farmer – their exploitation will be more effective and efficient – and the society in general – ICTs can help to reduce the environmental footprint caused by human activity in this productive sector.

Keywords

Efficiency, ICTs, irrigated agriculture, irrigation pivots, water.

"El acceso al agua potable y segura es un derecho humano básico, fundamental y universal, porque determina la sobrevivencia de las personas, y por lo tanto es condición para el ejercicio de los demás derechos humanos. Este mundo tiene una grave deuda social con los pobres que no tienen acceso al agua potable, porque eso es negarles el derecho a la vida radicado es su dignidad inalienable."

Carta Encíclica *LAUDATO SI'* del Santo Padre Francisco sobre el cuidado de la casa común.

"El auténtico desarrollo humano posee un carácter moral y supone el pleno respeto a la persona humana, pero también debe prestar atención al mundo natural y tener en cuenta la naturaleza de cada ser y su mutua conexión en un sistema ordenado."

Carta Encíclica *SOLLICITUDO REI SOCIALIS* del Santo Padre Juan Pablo II

Agradecimientos

En primer lugar, me gustaría dar las gracias a mi tutora, Lourdes, por permitirme desarrollar este Trabajo Fin de Master (TFM) en el que tanto interés tenía y por sus consejos y recomendaciones durante el pasado año.

En segundo lugar, a mi gran amigo Javi, pues en él surgió la idea central de este TFM. Igualmente, por toda la ayuda prestada no sólo durante este ajetreado año para mí, sino desde que nos conocimos allá por el año 2008, cuando ambos iniciamos los estudios I.T.T. en Sistemas de Telecomunicación en la Escuela.

En tercer lugar, a mi familia, especialmente a mis padres, mi hermana y mis abuelos. Por su apoyo y cariño incondicional durante estos años, para levantar el ánimo en momentos difíciles y para lograr que este momento se hiciera realidad.

Finalmente, y dado que con este TFM finalizo mi paso por la Escuela, que ha durado nada menos que siete años, a todos los profesores, compañeros de carrera y demás personal de la Escuela con los que he tenido la oportunidad de compartir ratos de aula, laboratorio, pasillo, biblioteca o cafetería.

Ha sido un placer,

Jesús Monge Álvarez.

Índice general

Índice de figuras

Índice de tablas

1

Introducción: contexto y objetivos

1.1 La importancia del agua para la vida

El agua, sustancia cuya molécula está formada por dos átomos de hidrógeno y uno de oxígeno (H_2O), es junto con la luz del sol y el oxígeno del aire, un elemento indispensable para el desarrollo de la vida y los ecosistemas [1].

Aunque el agua es uno de los elementos naturales más abundantes de la tierra, sólo el 2.53% del total es agua dulce, mientras que el resto es agua salada. En torno a un 60% de esta **agua dulce** se encuentra inmovilizada en forma de glaciales y nieves permanentes. Además, su **reparto por el globo terráqueo no es uniforme**: el continente americano concentra el 41%, Europa un 8%, África un 11%, Asia un 36% y Australia y Oceanía un 5%. Estos porcentajes entran en claro contraste con la población relativa de cada una de estas zonas. En este sentido, el continente asiático es el que está sometido a mayor presión, pues alberga el 60% de la población mundial. Por el contrario, el continente americano y Australia y Oceanía sólo alcanza el 14% y menos de un 1%, respectivamente. En el medio se sitúan Europa y África con un 13% cada una de ellas, si bien África, por su peculiaridades, es la más vulnerable al problema del agua [2], [3].

El **agua constituye una parte esencial de todo ecosistema**, tanto en términos cualitativos como cuantitativos. Una reducción del agua disponible ya sea en la cantidad, en la calidad, o en ambas, provoca efectos negativos graves sobre los ecosistemas. La madre naturaleza tiene unas capacidades limitadas de absorción y auto-limpieza. Si estas se sobrepasan, la biodiversidad se pierde, los medios de subsistencia disminuyen y las fuentes naturales de alimentos – por ejemplo, peces, hortalizas o frutas – se deterioran [2].

La principal forma de renovación del agua dulce son las precipitaciones, que a su vez constituyen la principal fuente tanto para el uso humano como para los ecosistemas. Las precipitaciones pasan al suelo y al subsuelo, así como se evaporan a la atmosfera mediante la evapotranspiración y alcanzan el mar por medio de ríos, lagos y humedales. Todos estos procesos son de vital importancia para el mantenimiento de bosques, prados de pastoreo y tierras de cultivo en secano.

Por todo esto, en los últimos años se ha dado valor e importancia a dos conceptos clave en relación al agua y los ecosistemas, a saber: que los ecosistemas no sólo poseen su propio valor intrínseco, sino que además proporcionan servicios esenciales al género

humano y, en segundo lugar, que la durabilidad de los recursos hídricos requiere una gestión participativa, desde una perspectiva holística del propio ecosistema [2].

Al desequilibrio natural ya mencionado anteriormente se unen dos nuevos factores de extrema incidencia: el **aumento de la población** y el **cambio climático** [4], [5].

A pesar de que el ritmo de crecimiento de la población mundial se ha ralentizado desde los años ochenta del siglo pasado, las cifras de población siguen creciendo rápidamente, sobre todo en los países en desarrollo [4]. Asimismo, derivado del desarrollo económico continuado – en particular de las economías emergentes – esta población en aumento exige una dieta cada vez más rica y variada (que cuente con más carne y productos lácteos, verduras y hortalizas, etc.), lo que aumenta todavía más la presión sobre la producción de alimentos y, en consecuencia, sobre el consumo de los recursos hídricos necesarios para su producción [2]. En este sentido, la gran mayoría de personas no son conscientes de la cantidad de litros de agua dulce y limpia que son necesarios para producir algunos de los alimentos más básicos. En base a los valores de referencia mostrados en la Tabla 1, si bien las necesidades de agua potable por persona son de 2 a 4 litros, son necesarios de 2000 a 5000 litros de agua para producir el alimento cotidiano que necesita una persona [6].

Finalmente, el aumento de población no solo supone un aumento de la demanda de alimentos sino también un aumento de la contaminación de los recursos de agua dulce. Una población en ascenso consume más energía y necesita de más productos y servicios. Ambas actividades, la producción de energía y la industria, son junto con la agricultura las que más agua dulce consumen y sus desechos, una fuente de contaminación directa. Véase el gráfico de la Figura 1, donde se recoge un desglose de las extracciones de agua por regiones y tipo de consumo. Unos 2 millones de toneladas de desechos son arrojados diariamente en aguas receptoras, incluyendo residuos industriales y químicos, vertidos humanos y desechos agrícolas (fertilizantes, pesticidas y residuos de pesticidas). Aunque los datos confiables sobre la extensión y gravedad de la contaminación son incompletos, se estima que la producción global de aguas residuales es de aproximadamente 1.500 km^3. Asumiendo que un litro de aguas residuales contamina 8 litros de agua dulce, la carga mundial de contaminación puede ascender actualmente a 12.000 km^3. Como siempre, las poblaciones más pobres resultan las más afectadas, con un 50% de la población de los países en desarrollo expuesta a fuentes de agua contaminadas [2].

Producto	Unidad	Consumo agua (m^3)
Trigo	Kg	1.2
Arroz	Kg	2.7
Maíz	Kg	0.45
Carne fresca de bovino	Kg	15
Carne fresca de oveja	Kg	10
Carne fresca de pollo	Kg	6
Cítricos	Kg	1
Legumbres, raíces y tubérculos	Kg	1

Tabla 1. Cantidad de agua requerida por unidad para los principales productos alimenticios. Claramente, el ganado es el que consume mayor cantidad de agua por unidad. Los cereales y cultivos como las legumbres, raíces y tubérculos consumen mucha menos agua [2].

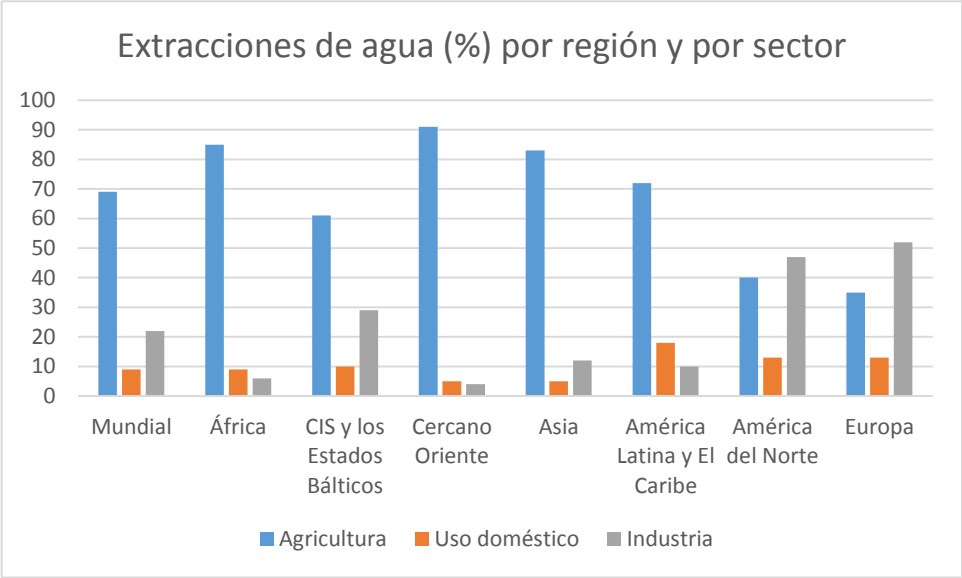


Figura 1. Gráfico porcentual con las extracciones de agua para cada región mundial según sectores. En términos mundiales, el peso relativo de la agricultura es claramente superior. Igualmente, destacar el contraste entre los países en vías de desarrollo y Europa o América del Norte [6].

Al mismo tiempo, el efecto preciso que el cambio climático produce sobre los recursos hídricos es incierto. Se prevé que las precipitaciones varíen respecto a su línea histórica según qué zonas del planeta (tanto en aumento como en decremento). Además la tendencia a condiciones meteorológicas extremas hará cada vez más frecuente fenómenos como sequías, avalanchas de lodo o inundaciones. Las estimaciones más recientes sugieren que el cambio climático será responsable de alrededor del 20% del incremento de la escasez global de agua [4].

El reconocimiento de estos desafíos ambientales ha aumentado el interés de instituciones internacionales, como la Organización de Naciones Unidas (ONU) [7] o algunas de sus agencias internas (FAO [8], *Food and Agriculture Organization* o UNESCO [9], *United Nations Educational Scientific and Cultural Organization*),

gubernamentales y no gubernamentales [10]. Así, en 1993 la Asamblea de Naciones Unidas declaró el **22 de Marzo como el Día Mundial del Agua** (Resolución 47/193). Posteriormente, la misma institución, en su Resolución 64/242 del 28 de Julio de 2010 proclamó el **derecho humano al agua y al saneamiento** y en su Resolución 65/154 del 20 de diciembre de 2010 proclamó **2013** como el **Año Internacional de Cooperación en la Esfera del Agua**. Finalmente, el periodo **2005-2015** se declaró **Decenio Internacional para la Acción «El agua fuente de vida»** (Resolución 58/217 del 23 de diciembre de 2003)**.**

Me gustaría terminar este subapartado con las palabras del actual Secretario General de la ONU, Ban Ki-moon, para conmemorar el Día Mundial del Agua de 2015 [11]:

«Para eliminar los múltiples problemas relacionados con el agua, debemos trabajar con un espíritu de cooperación urgente, **con mente abierta a las nuevas ideas y la innovación, y dispuestos a compartir las soluciones** que todos necesitamos para un futuro sostenible.»

1.2 El agua y la agricultura

La agricultura nació en el periodo Neolítico (4500-4000 a.C.) cuando las sociedades humanas se asentaron y se produjo una evolución desde economías basadas en la recolección, la caza y la pesca a economías en las que la producción de alimentos debía ser más constante para poder mantener la densidad de población [12].

Aunque a nivel mundial no existe un único punto de origen de la agricultura, pues su desarrollo se gestó en todo tipo de culturas, la zona de más influencia para España fue el denominado Creciente Fértil (Oriente Próximo desde Mesopotamia hasta el Antiguo Egipto). Desde este punto, las técnicas agrícolas y la sofisticación de los cultivos se fueron extendiendo por todo el mar Mediterráneo. Posteriormente, debido al devenir histórico de nuestro país, también recibimos influencias de las culturas precolombinas de América Central [12].

Tras su nacimiento, el mundo ya no volvió a ser el mismo. Si bien nadie pone en duda el hecho de que, para cualquier sociedad, una fuente constante de alimentos es algo positivo para su desarrollo, ciertos estudios antropológicos del fenómeno de nacimiento

y desarrollo de la agricultura subrayan algunos efectos negativos. Por ejemplo, la agricultura significó una reducción en la variedad de la dieta lo que a su vez provocó que el ser humano se volviera más vulnerable y dependiente de un determinado enclave. Aun así, todas las sociedades posteriores (Antigua Roma, Edad Media, Edad Moderna y Edad Contemporánea) siguieron necesitando de la agricultura para su subsistencia [13], [14].

Aunque un desarrollo de tantos años ha dado lugar a múltiples variantes, la agricultura se divide en dos tipos fundamentales: **secano** y **regadío**. La primera, se produce sin un aporte de agua por parte del agricultor. Es decir, se abastece únicamente de las lluvias o corrientes subterráneas. Por el contrario, en la agricultura de regadío – la de interés para este trabajo – el agricultor suministra las cantidades necesarias de agua a los cultivos mediante diversos métodos artificiales de riego, por muy artesanales que estos sean. Esta agua tiene una función estratégica, y no es otra que suplementar al agua de lluvia cuando esta no es suficiente para satisfacer las necesidades de los cultivos en áreas sujetas a excesiva variabilidad climática, o donde la producción de múltiples cultivos requiere de la provisión de agua fuera de la temporada de lluvias. Al asegurar la provisión de agua, el riego protege la producción agrícola y permite que los agricultores inviertan en un desarrollo agrícola más productivo [15].

La agricultura es el mayor usuario de agua dulce del mundo: 70% del consumo, si bien esta cifra puede alcanzar hasta un 90% en países subdesarrollados o en vías desarrollo [16]. La extensión de tierras de regadío en todo el mundo es de 227 millones de hectáreas aproximadamente, es decir, el 20% de todas las tierras cultivadas. Sin embargo, se espera que este porcentaje siga creciendo del mismo modo a como ya lo hizo durante la segunda mitad del siglo XX, cuando se expandió un 1.8% anual dando lugar a un aumento global de 130 millones de hectáreas [15].

También es importante destacar el **aumento de productividad que suponen los regadíos**. Las tierras bajo riego son de media tres veces más productivas que las tierras de secano. Es por esto que el 40% de la producción actual de alimentos proviene de tierras de regadío. Asimismo, se espera que este porcentaje crezca hasta un 50% en el año 2030, cuando un tercio de las tierras adicionales que se incorporen a la agricultura será regadas [15].

Por otro lado, **la agricultura es**, sobre todo a nivel local, **el epicentro de diversos sistemas económicos rurales** [2]. Si queremos alcanzar una economía verde,

entendiendo por economía verde aquella que mejora el bienestar y la equidad social a la vez que reduce significativamente los riesgos medioambientales [17], la agricultura debe jugar un papel esencial, dado que supone el 70% de consumo a nivel mundial y proporciona empleo al 40% de la población mundial. Además, el crecimiento del producto interior bruto generado por la agricultura es cuatro veces más efectivo a la hora de reducir la pobreza que el generado por otros sectores [16]. En este sentido, los efectos positivos de la inversión en sistemas de regadío son innegables. Por ejemplo, en la India la pobreza alcanza a un 69% de la población que vive en las zonas de secano mientras que la proporción desciende a un 26% en las zonas irrigadas [2].

En los países subdesarrollados o en vías de desarrollo, donde se da una agricultura extensiva de subsistencia, la vulnerabilidad de los agricultores en particular, y de toda la población en general, a la productividad agrícola es muy fuerte. De ahí que los regadíos cobren todavía más importancia. En este aspecto, se observa una diferencia clara respecto a la agricultura – de tipo industrial e intensivo – de los países desarrollados. En estos últimos, el acceso a la tecnología agrícola del siglo XX y a determinados productos complementarios (fertilizantes, pesticidas, abonos, semillas mejoradas, etc.) permitió aumentar la productividad agrícola sin recurrir a una extensión de las tierras de regadío de manera tan asidua. Véase el contraste entre las imágenes de la Figura 2. No obstante, esta tendencia se está invirtiendo con el paso de los años dado que muchas de estas tecnologías y productos pierden gran parte de su efectividad si la planta padece escasez de agua. Además, sus altos costes merman la rentabilidad económica de las explotaciones agrícolas [5].

Figura 2. Dos ejemplos del contraste entre la tecnología agraria empleada en los países subdesarrollados o en vías de desarrollo (izq.) y la de los países desarrollados e industrializados (dcha.).

Por todo esto, **la agricultura es el sector productivo en el que la escasez de agua o su uso eficiente tiene mayor relevancia.** Para tratar la escasez de agua en todos los sectores existen dos vías de acción fundamentales: aumentar el suministro o gestionar la demanda (reasignar los recursos hídricos dentro de cada sector y entre sectores). En el caso de la agricultura, parece mucho más sensata la segunda medida. Estas medidas se deben ver como un continuo desde la fuente de agua hasta el usuario final (el agricultor), y posteriormente, el consumidor de los productos agrícolas [18].

Aunque el término "eficiencia en el uso del agua" es un término paraguas que engloba múltiplos conceptos y actividades, la agricultura cuenta con tres mecanismos para gestionar la demanda de agua total dentro del dominio hídrico [18]:

- Reducir las pérdidas de agua
- Aumentar la productividad del agua
- Reasignar el agua

Con frecuencia se hace demasiado énfasis en la primera opción de las tres anteriormente mencionadas. En otras palabras, se dirigen todos los esfuerzos a reducir las pérdidas de agua de los sistemas de riego [18]. Sin embargo, esta medida de "eficiencia en el uso del agua" es poco efectiva cuando se emplean sistemas de canalización del agua por medio de acequias y riego por inundación (este tipo de regadío sigue existiendo tanto en países subdesarrollados como industriales). Un ejemplo de este tipo de canalización y técnica de regadío puede verse en la Figura 3. Estas dos técnicas de riego son muy efectivas pero poco eficientes, pues entre el punto de extracción y el de consumo se pierde mucha agua. Al mismo tiempo, es importante destacar que esta falta de eficiencia no tiene un impacto medioambiental excesivo dado que el agua perdida por grietas en las acequias, fugas, etc. vuelve al sistema hidrológico. Por ejemplo, por percolación en los acuíferos o como caudal de retorno de los sistemas fluviales. En conclusión, la proporción de agua que se pierde por evaporación o drenaje es muy baja. En estos casos, un aumento del rendimiento de los cultivos (producción por unidad de tierra) sería la vía más importante para aumentar la productividad de los cultivos con respecto al agua (mejorar la gestión de la tierra, rotaciones de cultivos, nuevas prácticas agronómicas, etc.).

Figura 3. Ejemplo del uso de un sistema de canalización por acequias para riego por inundación. La combinación de estas dos técnicas ofrece mucha efectividad pero poca eficiencia.

Por el contrario, cuando se trata de sistemas de regadío que emplean tecnologías modernas la consideración es completamente distinta. En la Figura 4 pueden verse dos ejemplos de estas tecnologías modernas para el manejo de los riegos. En estos sistemas, la canalización por medio de tuberías o gomas es mucho más fiable y, además, está mucho más controlada. O lo que es lo mismo, cualquier fuga que se produzca se puede detectar rápidamente y ser reparada sin demasiado esfuerzo. La principal desventaja de estos sistemas, aparte de su coste, radica que en que su gestión durante los periodos de funcionamiento obliga a personase *in situ* en la propia tierra de regadío. En más de una ocasión, esto suele acarrear numerosos desplazamientos de importancia (varios km). Igualmente, el desconocimiento de su rendimiento y/o estado durante los periodos de funcionamiento impide que estas **nuevas tecnologías de riego** no sólo faciliten la labor del agricultor sino que además **se conviertan en tecnologías para "la eficiencia en el uso del agua"**. Como se detallará en más profundidad en el siguiente subapartado, intentar dar una solución a este problema será el **principal objetivo del presente trabajo.**

Para terminar este subapartado indicar que, a pesar de las instituciones superiores (consejerías, ministerios, organismos internacionales, etc.) pueden jugar un papel muy importante en la lucha por un uso eficiente del agua en la agricultura, en última instancia es a nivel del agricultor como productor donde se lleva a cabo el mayor consumo de agua y donde más posibilidades de un cambio de gestión existen.

Figura 4. Dos ejemplos de tecnologías de riego modernas: pívot de riego (iza.) y cañón de riego (dcha.).

1.3 Objetivos y metodología

Hemos visto como el agua no es el único condicionante para la mejora de la producción y nutrición de las cosechas. Sin embargo, sin un acceso seguro al agua por parte de los agricultores las intervenciones sobre cualquier otro de los factores limitantes se verán mermadas o fracasarán.

Dado que los recursos hídricos se están reduciendo y que la competencia entre los distintos sectores se acentúa, la agricultura bajo riego necesita un análisis constante para discernir en qué forma la sociedad puede beneficiarse más efectivamente de su aplicación [15]. El acceso a un recurso natural tan básico como el agua debe ser compartido entre sus diferentes usos (agricultura, industria, energía, etc.) y entre sus diferentes usuarios.

Actualmente, el uso eficiente del riego se sitúa en torno a un 38% en todo el mundo, pero debería mejorar lentamente hasta alcanzar un promedio del 42% en 2030 [2]. Para lograr este objetivo, la **modernización de la agricultura será esencial** si queremos asegurar que la entrega de agua a los agricultores sea más flexible y más confiable y para equilibrar la eficiencia y la equidad en el acceso a la misma. Esto requiere **inversiones con objetivos bien dirigidos a la modernización de la infraestructura**, la reestructuración institucional y al **mejoramiento de la capacidad técnica de los agricultores** y/o administradores del agua [15].

Conjuntamente, este proceso de modernización no sólo será beneficioso para los agricultores sino también para los ecosistemas en los que estas explotaciones agrícolas se encuentran. El impacto negativo del manejo del agua en la agricultura está relacionado con el uso de la tierra y el agua, especialmente cuando se abusa de los ecosistemas

naturales por la extracción de agua, la erosión y la pérdida de biodiversidad del suelo. La difusión de enfermedades relacionadas con el agua y la degradación del suelo por inundaciones o salinización son también el resultado de una planificación y manejo inadecuados del riego y el drenaje agrícola [15]. Es esencial, por lo tanto, encontrar caminos alternativos para que la agricultura pueda aliviar este impacto y mantener la integridad y la productividad de los ecosistemas de los cuales depende.

Siguiendo esta misma línea, el Panel de Expertos de la segunda reunión preparatoria del Comité de la Conferencia de las Naciones Unidas sobre el Desarrollo Sostenible de 2011 emitió la siguiente conclusión [19]:

«La transición hacia una economía verde implica una verdadera **revolución tecnológica** que tendrá un profundo impacto en las **estructuras productivas** y en los patrones de consumo.»

En el mismo foro, también se dijo que **las Tecnologías de la Información y las Comunicaciones (TIC) constituyen un factor clave para la economía verde en todos los sectores. El empleo de las TIC** no sólo facilita el acceso de la información a todo el mundo sino que permite reducir el impacto medioambiental y **ayuda a encontrar soluciones encaminadas a mejorar la eficiencia** de muchos aspectos como el consumo de combustible, las redes de distribución eléctrica y, también, el **consumo de agua** [19].

Mediante las tecnologías se impulsará la eficiencia hídrica, energética y se contribuirá a alcanzar los Objetivos de Desarrollo del Milenio en la construcción de una economía verde. Las tecnologías innovadoras en el campo de la agricultura posibilitarán un uso más eficiente de los recursos hídricos [19]. Asimismo, en otro informe de la FAO [18], se recomendó **la búsqueda de soluciones con especificidad según el contexto**. Es decir, que aunque el problema de un uso eficiente del agua en la agricultura es un problema global, no tiene una solución global. Esto conlleva que tendrá que ser cada país, región, comarca o incluso agricultor el que adquiera y/o desarrolle las tecnologías que considere necesarias para controlar y administrar eficientemente el consumo de agua en sus explotaciones agrarias.

En resumen, la agricultura, y en particular la agricultura de regadío, está sufriendo cambios muy rápidos y enfrentándose a viejos y a nuevos problemas. Los agricultores de todo el mundo habrán de adaptarse a los nuevos mercados internacionales, a la globalización y a la conexión e interdependencia entre la producción y los patrones de

consumo de las personas. En este nuevo entorno, el progreso tecnológico es sin duda la mejor herramienta para lograr progresos agronómicos y alcanzar así una economía verde que sea capaz de dar de comer a todos los seres humanos sin destrozar el planeta en el que vivimos.

Por todas las razones expuestas a lo largo de esta introducción, se considera que cualquier nuevo proyecto científico desarrollado dentro de este contexto **cumple con los requisitos de originalidad y relevancia.** De este modo, el objetivo principal del trabajo realizado para este Trabajo Fin de Master (TFM) será:

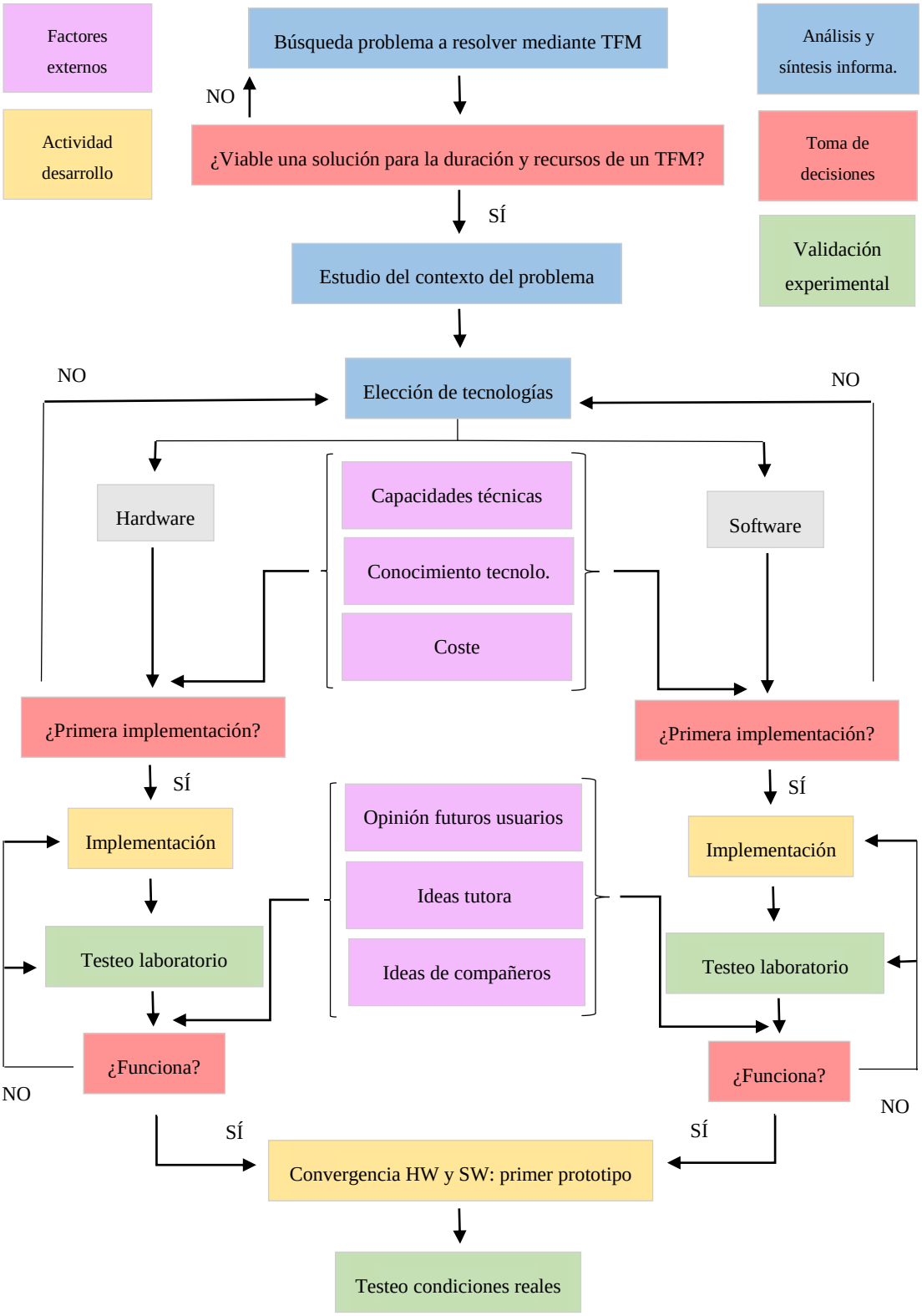
Diseñar y, en la medida de lo posible, implementar un prototipo que constituya una ***INNOVACIÓN INCREMENTAL*** *de alguna de las tecnologías de riego existentes, permitiendo su control a distancia en aras de una maximización de la eficiencia del riego y de evitar desplazamientos innecesarios. En concreto, este TFM se centrará en los pívots de riego (Figura 4, imagen de la izquierda).*

Para lograr el objetivo anterior se espera que la solución aportada tenga las siguientes características:

- El control a distancia se realizará por medio de una interfaz web
- Esta interfaz web deberá ofrecer un grado de usabilidad adecuado tanto en dispositivos fijos (PC) como móviles (portátiles, smartphones, etc.).
- El sistema deberá ser capaz de adquirir y procesar los datos que se consideren necesarios para informar al usuario de su rendimiento.
- El sistema contará con un mecanismo de alerta que informe de manera inmediata cuando se produzca algún evento considerado de riesgo (pérdida de presión, rotura, desviación, etc.).
- Considerando el carácter tan aplicado y práctico de este TFM, no se puede plantear como objetivo el ofrecer una mera solución tecnológica en bruto, sino que se debe tener en cuenta la eficiencia de la propia solución. Por ejemplo, el consumo de energía del prototipo (*a priori*, todo parece indicar que éste será despreciable respecto al de los otros sistemas con los que trabaje de manera coordinada; Por ejemplo, el motor de propulsión del pívot, la bomba para el agua, etc.) o el coste de un control a distancia – es decir, el coste de emplear alguno de los sistema de comunicación comerciales para poder controlar el pívot a distancia. Todo esto aumentará las posibilidades de que el prototipo pueda llegar a convertirse en una

TRANSFERENCIA TECNOLÓGICA, y en consecuencia, en un producto comercial accesible para la totalidad de sus potenciales usuarios.

Para lograr estos objetivos, se propone el siguiente plan de trabajo:



2

Tecnologías del desarrollo

En este capítulo se expondrán las características de las principales tecnologías empleadas en el desarrollo del prototipo, tanto hardware como software, y se justificará su elección en base a distintos factores o criterios.

2.1 El hardware base: Raspberry Pi

Tras considerar viable el proyecto planteado en la Introducción según el criterio de disponibilidad de tiempo que fija el TFM del MUI-TIC, había que determinar qué tecnologías se emplearían en su desarrollo de modo que también se cumpliera el criterio de disponibilidad de recursos así como el mayor número de características posibles de la solución planteada (apartado 1.3).

Inicialmente, se barajaron dos posibles vías de desarrollo:

1) Modificar la tecnología de fábrica que llevan los pívots de modo que cumpla con los requisitos y características fijados.
2) Complementar la tecnología de fábrica que llevan los pívots para que, de manera conjunta con el/los dispositivo/s adheridos, cumpla con los requisitos y características fijados.

La primera opción de las dos anteriormente mencionadas tiene una desventaja clara: la necesidad de disponer o tener acceso a un pívot de regadío durante todo el proceso de desarrollo del TFM. Es decir, no cumple el criterio de disponibilidad de recursos. Además, también hay otras desventajas a destacar. Teniendo en cuenta que en el objetivo principal de este TFM es el desarrollo de un prototipo que constituya una innovación incremental, el modificar la tecnología existente no parece la mejor forma de conseguirlo. Hay varias razones que apoyan esto. Si uno se fija en otras innovaciones incrementales, por ejemplo, el incluir cámaras de fotos en los teléfonos móviles o navegadores GPS (*Global Positioning System*) en los coches, observa como la innovación no supuso una modificación sustancial de la tecnología base que se pretendía mejorar, si bien siempre hay que adaptarla. En otras palabras, un teléfono móvil siguió siendo un dispositivo cuya misión principal era llamar y recibir llamadas o un coche se puede seguir fabricando y vendiendo sin navegador GPS si el cliente así lo solicita. En segundo lugar, el hecho de que no exista un formato universal o estandarizado de pívot de regadío complica mucho

este enfoque. Esto supondría que la solución desarrollada sólo sería válida para el formato o modelo empleado y por lo tanto, se estaría perdiendo parte de la **generalidad** que todo trabajo **científico o tecnológico** debiera poseer así como limitando las posibilidades de una futura transferencia tecnológica. Finalmente, resaltar la incomodidad de trabajar con un dispositivo de tan grandes dimensiones y la ausencia de garantías de disponer de las herramientas y material de trabajo adecuado.

Por consiguiente, **la opción de complementar la tecnología existente con un nuevo dispositivo parece la más coherente.** Ahora, habría que determinar cuál sería la base del nuevo dispositivo. Tras una búsqueda rápida en la literatura científica y en los productos de mercado, surgieron las siguientes tres opciones:

1) Raspberry Pi
2) Arduino
3) Diseñar y fabricar de manera *ad-hoc* una placa

La tercera opción fue la primera en quedar descartada pues, como se verá a continuación, ya existían en el mercado tecnologías suficientemente desarrolladas como para cumplir los requisitos y características planteados. El haber diseñado y fabricado de manera *ad-hoc* una nueva placa hubiera supuesto "rehacer la rueda", al menos parcialmente, metodología que *a priori* es mejor evitar. Por lo tanto, ya sólo quedaba decidir entre una de las otras dos opciones: Raspberry Pi o Arduino.

Tanto Raspberry Pi como Arduino fueron diseñadas originalmente para ser herramientas de enseñanza, es por ello que se han vuelto tan populares – ambos dispositivos son muy fáciles de aprender a usar. En la Figura 5, se puede ver una foto de cada uno de estos dispositivos. Dar una respuesta absoluta a si es mejor Arduino o Raspberry Pi es algo muy complicado, pues depende enormemente de la subjetividad del que habla y del uso que se les quiera dar. Es una pregunta equivalente a otras como: ¿Es mejor un Mac o un PC?, ¿Instalo Windows o instalo Linux?, ¿Instalo Ubuntu o instalo Fedora?, etc. Por ello, a continuación se hará una comparativa entre ambas, definiendo las principales características y virtudes de cada uno de ellas en relación al objetivo principal del proyecto.

Figura 5. Imagen de una placa Arduino (izq.) y de una Raspberry Pi modelo B (dcha.).

Aunque ambas plataformas de hardware pueden resolver problemas similares, en realidad, son muy diferentes. La primera y más grande diferencia reside en su ser mismo: Raspberry Pi es un ordenador completamente funcional [20], mientras que Arduino es sólo un microcontrolador [21], es decir, un componente muy particular de un ordenador.

Derivado de lo anterior, surgen otra diferencia fundamental: la primera puede ejecutar todo un sistema operativo mientras que la segunda sólo pequeñas aplicaciones en C o C++. Esta diferencia ya entra en relación con algunos de los objetivos y características propuestos. Como se dijo en la Introducción, el prototipo debe ir acompañado de una interfaz web que permita su manejo a distancia, evitando así desplazamientos innecesarios. La carga y ejecución de esta plataforma web será un proceso mucho más sencillo en la Raspberry Pi que en Arduino.

Al ser un ordenador, la Raspberry Pi lleva incorporados otros elementos de hardware de los que Arduino carece [22], [23]. La Tabla 2 recoge los principales componentes de los que consta cada uno y otras características como tamaño, precio aproximado, etc.

Como se puede ver en la Tabla 2, Arduino sólo dispone de un puerto USB, lo que limita mucho nuestra capacidad para interconectar otros dispositivos, como por ejemplo, un receptor GPS. Respecto a coste y precio, ambas placas son muy parecidas y ambas cuentan con entornos de desarrollo.

En relación al sistema operativo, la placa Raspberry acepta cualquier distribución de Linux, lo que es un gran partido dado que es libre y de código abierto [24]. De entre las muchas distribuciones existentes, existe una especial para la Raspberry Pi denominada Raspbian [25]. Raspbian no es otra cosa que una versión optimizada para la arquitectura de hardware de la Raspberry Pi basada en Debian.

Componente	Arduino	Raspberry Pi modelo B
Precio en dólares USA	30	35
Tamaño (cm)	7.6 x 1.9 x 6.4	8.6 x 5.4 x 1.7
Memoria (MB)	0.002	512
Velocidad de reloj (MHz)	16	700
Tarjeta de red	No dispone	10/100 Ethernet RJ45
Multitarea	No	Sí
Voltaje de entrada (V)	De 7 a 12	5
Memoria flash	32 KB	Tarjeta SD (de 2 a 16 GB)
Puertos USB	Uno	Dos
Sistema Operativo	Ninguno	Todas las distribuciones de Linux
Entorno de desarrollo	Sí	Cualquier con soporte Linux (IDLE, etc.)

Tabla 2. Principales componentes de las plataformas de hardware Arduino y Raspberry Pi modelo B.

Adicionalmente al propio sistema operativo, la distribución Raspbian incluye casi 35000 paquetes de software extra pre-compilados, lo que aumenta enormemente las posibilidades de explotación de esta placa de hardware y reduce potencialmente nuestro tiempo de desarrollo [25]. De la misma forma a como ocurre con la práctica totalidad de las distribuciones basadas en Linux, la comunidad de desarrolladores y usuarios cuenta con foros, blogs y otros sitios webs donde se comparten ideas y códigos.

La diferencia en la velocidad de reloj es clara, pero por otro lado, totalmente lógica. Arduino tiene como elemento principal un microcontrolador, es decir, un componente electrónico de propósito específico pensado para ejecutar una única función a la vez. Por el contrario, la Raspberry Pi cuenta con un microprocesador de propósito general con **capacidad multitarea** – puede ejecutar varios programas en segundo plano mientras está activado [22], [26]. Esta característica puede ser de suma importancia para darle un valor añadido a nuestro prototipo, especialmente en lo que respecta a la **escalabilidad.**

Lo anterior podría llevar a pensar que Arduino es menos útil, pero nada más lejos de la realidad. Arduino es ideal para pequeños proyectos y prototipos electrónicos, pues contiene diferentes pines de entrada y de salida que permiten conectar una infinidad de sensores y actuadores. Sin embargo, en cuanto al prototipo o proyecto se le exige algo más que el mero procesamiento de la información recogida por los sensores (en nuestro caso por ejemplo, la aplicación web, el posible uso de GPS, etc.), sus deficiencias quedan patentes. En este mismo sentido, la Raspberry Pi se diferencia de una computadora normal no sólo en su precio y tamaño, sino también por su capacidad de integración con proyectos de electrónica. Por ejemplo, se pueden controlar LEDs, dispositivos de corriente alterna, así como sensores de varios tipos.

Ninguna de las dos placas cuentas con capacidad de almacenamiento, pero en el caso de la Raspberry Pi, se le pueden conectar **tarjetas SD** de distinta capacidad. Estas tarjetas SD pueden funcionar como **memoria flash** de todo el sistema, lo que permite cambiar rápidamente de versión de sistema operativo o actualizar el software [23]. Es decir, se dota al prototipo de **modularidad** para poder actualizarlo o modificarlo sin necesidad de cambiar aquellos otros componentes que no se vean afectados.

Respecto a las capacidades de conexión de cada placa, la Tabla 2 muestra que sólo la Raspberry Pi tiene incorporado un puerto Ethernet. El disponer de un puerto Ethernet no es una característica indispensable del prototipo, pues durante su funcionamiento no se utilizaría. No obstante, si es algo que facilita enormemente las tareas de desarrollo y/o mantenimiento de la parte software del prototipo. Gracias al puerto Ethernet y sus capacidades de alto nivel (sistema operativo, multitarea, etc.) la Raspberry Pi se puede dejar conectada a un router o toma de red y acceder a ella remotamente (VPN, FTP) para probar nuevos códigos o aplicaciones. De este modo, una vez que la parte hardware del prototipo estuviera desarrollada, el desarrollo de la parte software se podría llevar a cabo desde cualquier parte.

Esta característica también es útil desde la perspectiva del usuario final. Si finalmente el prototipo llegara al mercado, el usuario final sólo tendría que conectar el producto a alguna toma de red, la de su propio domicilio por ejemplo, para actualizar su software o para recibir asistencia remota en caso de que alguno de los componentes pueda estar fallando, por ejemplo, testeo de alguno de los pines de conexión con los sensores o actuadores, etc. En resumen, el puerto Ethernet no es un elemento imprescindible de nuestro prototipo pero le da un valor añadido potencialmente muy útil tanto para desarrolladores como usuarios finales.

Los requisitos de alimentación de cada placa de hardware son diferentes. La Raspberry Pi requiere una alimentación constante de 5 voltios para permanecer encendida. Además, el apagado de la Raspberry Pi debe llevarse a cabo mediante un proceso software de manera equivalente a como se hace con cualquier ordenador. Por otro lado, Arduino ofrece algo más de flexibilidad en el voltaje de alimentación, entre 7 y 12 voltios, y se apaga directamente desde uno de sus interruptores o pines. Dado que en nuestro caso no va haber problemas en disponer de una fuente de alimentación eléctrica, esta característica de ambas placas parece irrelevante. Sin embargo, si lo anterior no se cumpliera, la alimentación y portabilidad del prototipo constituiría un problema, pues no

se pueden alimentar mediante una simple pareja de pilas AA. Habría que buscar algún otro sistema basado en batería, algo que por otra parte no parece complicado.

En cuanto a los sensores con los que pueden interactuar, la conexión de sensores analógicos es mucho más fácil en Arduino, pues el microcontrolador es un dispositivo pensado para responder rápidamente según el código cargado en él a una amplia gama de datos procedentes de los sensores. Este enfoque es ideal si una serie de comandos se repite constantemente o responde a los datos de los sensores ajustando el comportamiento de otros dispositivos o actuadores, que – parcialmente – es lo que nosotros buscamos [22]. En el otro extremo, la Raspberry Pi requiere un software especial para interactuar eficazmente con este tipo de sensores, algo que no siempre es deseable. Para solucionar esto, se podrían utilizar ambas placas, siendo la Raspberry Pi la placa base que emite las órdenes a Arduino, que las ejecuta y le devuelve *feedback* a la Raspberry Pi con la información de los sensores [23], [27]. Con todo, este planteamiento encarecería el coste final del prototipo, por lo que será evitado.

A pesar de las virtudes y deficiencias de cada placa, ambas son excelentes herramientas para el desarrollo de pequeños proyectos de electrónica, ingeniería, etc. El componente principal de Arduino es un microcontrolador cuyo firmware interpreta líneas de código simple. Su propósito principal es interactuar con otros sensores y dispositivos, por lo que es ideal para proyectos de hardware en los que sólo se quiere dar respuesta ante distintas lecturas de sensores. Esto le dota de una capacidad de procesamiento – analógico, especialmente – en tiempo real a la que la Raspberry Pi no alcanza. Por el contrario, el incorporar un sistema operativo aporta a la Raspberry Pi capacidad multitarea y la posibilidad de ofrecer servicios de alto nivel al usuario (interfaz web). Además, puesto que utiliza una variante de Linux, la Raspberry Pi se beneficia de las tres décadas de desarrollo que éste lleva a sus espaldas. En una comparativa más tecnológica y menos funcional de cada uno, la Tabla 2 deja claro que la Raspberry Pi es una placa mucho más completa: tiene una frecuencia de reloj 40 veces superior a Arduino, 128000 veces más de memoria RAM, capacidad de incorporar almacenamiento externo (tarjetas SD) y mayores capacidades de interconexión (dispone de tarjeta de red Ethernet y mayor número de puertos USB). Finalmente, indicar que ambas placas tienen grandes y activas comunidades de usuarios, entornos de desarrollo integrado que facilitan la elaboración de código para las mismas y su tamaño y precio son similares.

En conclusión, la simplicidad de Arduino hace que éste sea una apuesta mucho mejor para proyectos exclusivamente de hardware, mientras que la Raspberry Pi brilla por su capacidad de ofrecer las ventajas del software de alto nivel (Sistema Operativo) junto con el manejo de dispositivos como sensores o actuadores.

Como se ha dicho al principio, la elección de uno u otro depende en gran medida de que es lo que se quiera lograr con ellos. Considerando el objetivo de este TFM y las características deseables de la solución propuesta para lograr el mismo (véase apartado 1.3), parece que la elección más coherente es la placa Raspberry Pi. Para contrarrestar algunas de las deficiencias inherentes a la misma y que acabamos de explicar, con algunas de las deficiencias inherentes a la misma que acabamos de explicar, se utilizara hardware y/o software adicional que será descrito en los siguientes subapartados de este capítulo.

2.2 Otros elementos del hardware

Como se ha explicado en el subapartado anterior, una de las principales desventajas de la placa Raspberry Pi es la dificultad para tratar con sus entradas, especialmente las de tipo analógico. Para subsanar este problema y otros de menor importancia, se recurrirá a cierto hardware adicional que se describirá a continuación.

2.2.1 Custard Pi 2: General Purpose I/O for the Raspberry Pi

La placa Custard Pi 2 no tiene otra misión que la de facilitar el manejo de las entradas y salidas de propósito general (GPIO, *General Purporse Input/Output*) con las que cuenta la Raspeberry Pi para el manejo de dispositivos externos. Al mismo tiempo, la Custard Pi 2 protege a la Raspeberry Pi de sufrir cualquier tipo de daño debido a la conexión de alguna señal de voltaje inadecuado en las GPIO [28].

Esta placa es uno de los complementos de las Raspberry Pi más comunes y se puede encontrar en numerosas tiendas on-line de electrónica a precios asequibles. Generalmente, la placa se vende con los componentes sueltos. El usuario, simplemente tiene que soldarlos y verificar las conexiones. En la Figura 6 se puede ver una placa Custard Pi 2 tras el proceso de montaje.

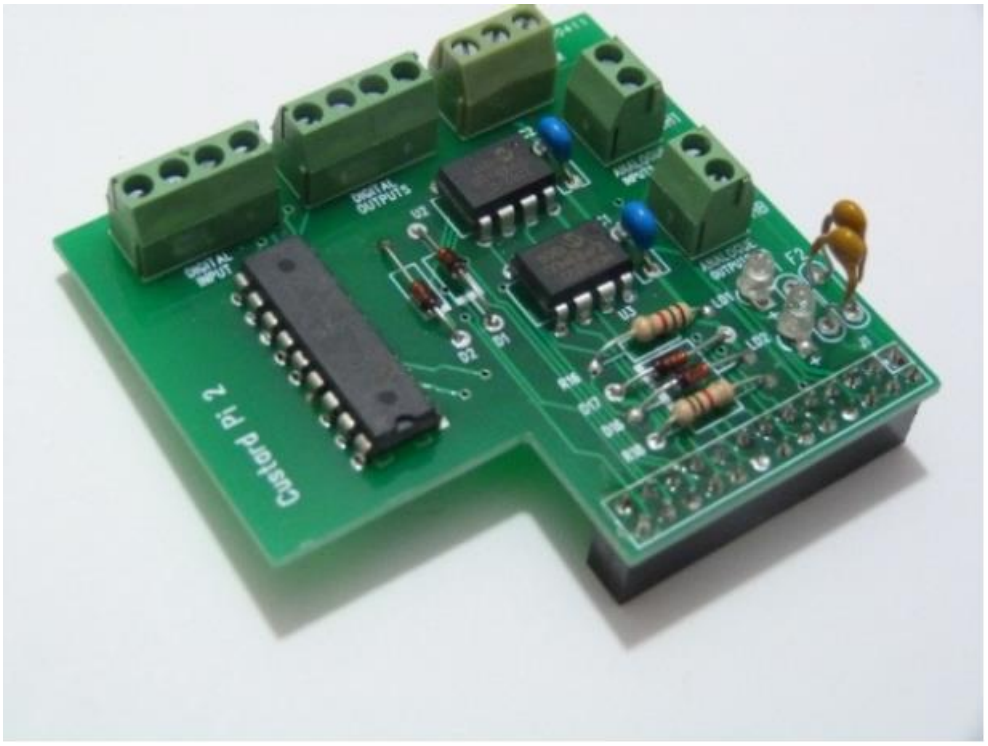


Figura 6. Placa Custard Pi 2 tras el proceso de soldadura de sus componentes.

En concreto, las Custard Pi 2 permite el manejo de las siguientes entradas/salidas:

- Dos entradas analógicas de 12 bits
- Dos salidas analógicas de 12 bits
- Cuatro entradas digitales
- Cuatro salidas digitales

La conexión de la placa a la Raspberry Pi es directa, por medio de los 26 pines que esta dispone como conectores GPIO. Los LEDs que incorpora se enciende para unos voltajes de entre 3.3 V y 5.5 V, permitiendo así verificar un correcto montaje y conexión de la misma. Asimismo, como se observa en la Figura 6, las doce entradas analógicas o digitales cuentan con terminales de tornillo, lo que facilita la conexión y desconexión de los dispositivos externos, e igualmente para el caso de los terminales de alimentación.

Destacar que, de entre los 26 pines de conexión con las Raspberry, hay seis que no están protegidos contra sobretensiones mediante los correspondientes fusibles. El nivel lógico más común es de 3.3 V. Sin embargo, admite hasta 20 V en sus entradas digitales y hasta 3.3V en sus entradas analógicas. Finalmente, la Custard Pi 2 dispone de otros tres pines, uno a 3.3 V, otro a 5.5 V y otro a tierra (sin protección, para alimentar otros dispositivos externos como sensores, actuadores, etc. [29], [30].

Si el montaje se ha realizado correctamente, inmediatamente después de conectar la placa a las Raspberry Pi y alimentarlas, ambos LEDs se encienden.

En último lugar, indicar que, junto con los componentes de la placa, también se permite al cliente que los adquiere descargarse cuatro scripts en código Python que permiten verificar rápidamente el estado de las entradas y salidas analógicas y digitales (previa carga de los scripts en las Raspberry Pi, obviamente) [30].

Por lo tanto, se puede concluir que la Custard Pi 2 es una solución eficaz y barata a la que podría ser considerada la principal desventaja de la Raspberry Pi respecto a otras alternativas. En la Figura 7 se pueden ver una Raspberry Pi en modo básico y la misma con la placa Custard Pi 2 conectada.

2.2.2 Hardware complementario

La Raspberry Pi en adhesión con la placa Custard Pi 2 constituirá el hardware básico del prototipo final. No obstante, también se adquirieron pequeños elementos de hardware complementario para facilitar las tareas de desarrollo. En ellos, destacan:

- Contector USB-Wifi: aunque la Raspberry Pi cuenta con entrada de cable Ethernet, gracias a este pequeño dispositivo se puede conectar a cualquier red Wifi como cualquier otro ordenador o, en su defecto, si las Raspberry Pi se conecta a la red mediante cable, generar una red Wifi propia que nos facilite a nosotros como desarrolladores la carga y descarga de códigos, aplicaciones, etc.
- Tarjeta de memoria SD: su función es la de almacenar el sistema operativa y el código fuente de la aplicación web.

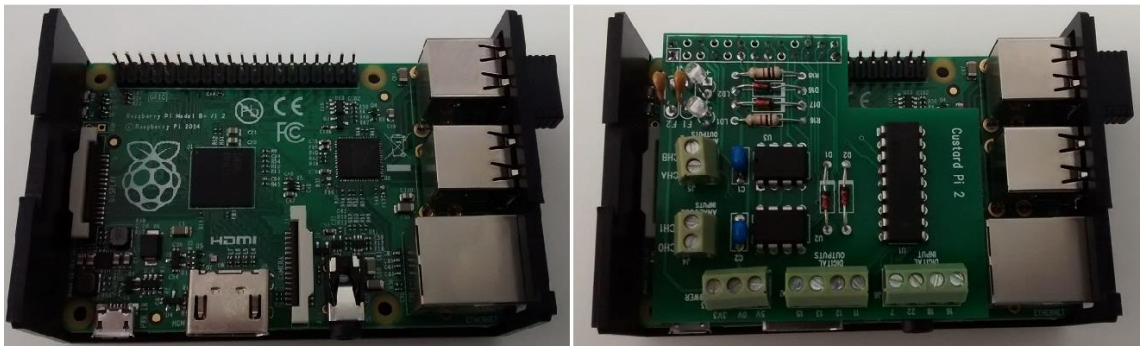


Figura 7. Raspberry Pi (izq.) y Raspberry Pi con la placa Custard Pi 2 incorporada (dcha.) para facilitar el control de las GPIO.

2.3 La aplicación web en PHP

En este subapartado se pretende decidir en qué tecnología implementar la aplicación web que se incorporará en el hardware compuesto por la Raspberry Pi y la placa Custard Pi 2 para facilitar el manejo remoto del pívot de riego.

Como cualquier aplicación web, su base será el lenguaje HTML (*HyperText Markup Language*) y CSS (*Cascading Style Sheets*). Sin embargo, estas tecnologías son tan básicas y extendidas que no se abordará una explicación de las mismas para no sobrecargar esta memoria. Sobre HTML y CSS se pueden utilizar multitud de lenguajes que permiten el desarrollo de páginas webs dinámicas. Algunas de las opciones son:

- PHP
- JavaServer Pages (JSP)
- Python
- Ruby

A continuación, se analizarán las ventajas y desventajas que cada una de ellas tiene en relación al objetivo y características de la solución propuestas en este TFM.

2.3.1 PHP

PHP se constituye como un lenguaje de programación interpretado (tipo *script*) que se ejecuta del lado del servidor y que desde su nacimiento se concibió para la creación de páginas web dinámicas, aunque en la actualidad se emplea también para otros propósitos. Es un lenguaje de propósito general en el que se pueden programar aplicaciones de todo tipo y tiene una licencia propia avalada por la *Free Software Foundation* [31].

Su código se puede incrustar en las páginas HTML pero no es visualizado por el navegador ni hace falta descargarlo cada vez que se ejecuta. De este modo, **la programación en PHP es segura y confiable.** Además, posee conectividad la mayoría de motores de bases de datos que se utilizan en la actualidad. A pesar de que esta característica no es necesaria en este momento, podría resultar de utilidad en un futuro. Por ejemplo, si se quiere que una misma aplicación web permita controlar varios pívots de riego [32].

Para poder trabajar con él únicamente se necesita un servidor Apache o IIS (*Internet Information Services*). En la actualidad, el lenguaje es mantenido por una comunidad pública que ofrece documentación y soporte a través del sitio web oficial.

2.3.2 Java Server Pages (JSP)

En realidad, JSP no constituye un lenguaje de programación en sí mismo, sino que es una tecnología derivada del lenguaje de programación Java que permite a los desarrolladores crear código dinámico. JSP no está licenciado como software libre, pues es un desarrollo de la compañía Sun Microsystems, actualmente propiedad de Oracle Corporation [33].

Para poder ejecutarlo, es necesario un servidor web que disponga de un sistema operativo con una máquina virtual Java y los *servelets*, que es una clase del lenguaje de programación Java utilizada para ampliar las capacidades de un servidor. Igualmente, dispone de una librería de etiquetas que implementa funcionalidades de propósito general en aplicaciones web: JSTL (*JSP Standard Tag Library*).

Para programar en JSP es recomendable tener algo de experiencia con la programación orientada a objetos en general y con Java en particular. Una vez dominado esto, se puede ir aprendiendo poco a poco el sistema de *servelets*.

JSP, al derivar de Java, hereda muchas de sus ventajas. Como Java es un lenguaje de propósito general, JSP excede con mucho las capacidades necesarias para el mundo web. Esto permite llevar las aplicaciones web un paso más allá, separando en distintos niveles la aplicación web y dejando sólo a JSP la responsabilidad de generar el código HTML dinámico. Del mismo modo, hereda la portabilidad de Java, por lo que una aplicación en JSP se puede ejecutar en múltiples plataformas sin cambios. Finalmente, destacar su robustez y la excelente gestión de errores.

La persistencia de Java también está presente en JSP. Básicamente, esta característica consiste en que cada vez que un código JSP recibe una petición se comienza a ejecutar en su propio hilo. Sin embargo, si más adelante el mismo JSP recibe otra petición, no volverá a comenzar su ejecución sino que persiste de una ejecución a la siguiente [34]. Esto hace que sea un lenguaje muy eficiente para conexión a bases de datos o manejo de sesiones de usuario, por ejemplo.

2.3.3 Python

Ya se ha hablado de este lenguaje en el subapartado relativo al hardware de este capítulo, pues la placa Custard Pi 2 se ofrece con cuatro scripts en este lenguaje para el testeo de los GPIO tanto analógicos como digitales [35].

Python es un lenguaje de programación con una amplia variedad de aplicaciones. Como PHP, es un lenguaje interpretado (*script*) multiparadigma – soporta programación orientada a objetos, imperativa y funcional. Es multiplataforma.

Es administrado por la Python Software Foundation y posee una licencia de código abierto. Python fue creado buscando una sintaxis limpia que favoreciera la legibilidad del código. Esto es lo que se conoce como "filosofía Python". Funciona en los tres principales sistemas operativos (Linux/Unix, Window, MAC OC X) o alguna de sus variantes [35].

Como principal desventaja, su lentitud pese a ser un lenguaje interpretado.

2.3.4 Ruby

Ruby es un lenguaje de programación interpretado, de muy alto nivel y orientado a objetos, cuya implementación oficial se distribuye bajo licencia de software libre. Su creador, el programador japonés Yukihiro Matsumoto, buscaba un lenguaje con las mejores características de Python y Perl. Así, recopilo aquellas partes que más le gustaban y creó un nuevo lenguaje cuyo principio de diseño es el desarrollador y no la máquina. Es decir, un lenguaje más natural que productivo (que se adaptara mejor a la forma de pensar de una persona que de una máquina) [36].

De este modo, se evita cualquier confusión y se minimiza el trabajo del programador. Aunque esto pueda parecer algo beneficioso para cualquiera, no todos los programadores están a favor de ello pues se alega que con experiencia en otros lenguajes sin este enfoque (lo que incluiría prácticamente todos), adaptarse a Ruby no es tarea sencilla [36].

Ruby es multiplataforma, dado que opera con la mayoría de sistemas operativos. Como le ocurría a Python, su principal desventaja es su lentitud a pesar de ser interpretado.

En la Tabla 3, se recoge una comparativa de los cuatro lenguajes.

Característica	PHP	JSP	Python	Ruby
Multiplataforma	Sí	Sí	Sí	Sí
Paradigma	Multiparadigma	Orientado a objetos	Multiparadigma	Orientado a objetos y reflexivo
Gratuito	Sí	No*	Sí	Sí
Integración BBDD	Sí**	Sí	Sí	Sí
Dificultad	Baja***	Alta***	Media	¿?
Rendimiento	Alto	Bajo	Medio	Medio
Soporte en Internet	Sí	Sí	Sí	Sí
Plataforma de desarrollo	Sí (L/W-AMP) [37]	Sí (Apache Tomcat) [38]	Sí (Django) [39]	Sí (Ruby on Rails) [40]

* Aunque fue creado y mantenido por una empresa, se puede encontrar en Internet alternativas gratuitas para desarrollar páginas webs dinámicas en JSP.

** Muy buena integración con Apache+MySQL, pero no está estandarizado al ser un lenguaje de creación comunitaria.

*** PHP, muy fácil de aprender para programadores que conocen C o C++; Por el contrario, para JSP se necesita conocer Java.

Tabla 3. Tabla comparativa de los cuatro lenguajes de programación candidatos para el desarrollo de la aplicación web.

2.3.5 Conclusión

A lo largo del subapartado 2.3 se han analizado las virtudes y defectos de cuatro lenguajes de programación con los que se podría desarrollar la aplicación web asociada al prototipo.

En términos estrictamente tecnológicos, cualquiera de los cuatro constituiría una elección coherente y fiable para el desarrollo de la aplicación web. Sin embargo, todo proyecto se desarrolla dentro de un contexto determinado y con unos objetivos específicos, de modo que serán ciertos matices entre unos y otros los que decanten la elección.

El hecho de que los cuatro sean multiplataforma es algo positivo pues nos permitiría modificar el sistema operativo de la Raspberry Pi sin necesidad de cambiar nada en la aplicación web para el control remoto.

Respecto al paradigma de cada uno, PHP y Python son los únicos con un enfoque multiparadigma, mientras que JSP y Ruby están más centrados en el paradigma orientado a objetos. Esta característica pasaría a ser irrelevante siempre y cuando el desarrollador de turno conozca el paradigma del lenguaje de programación seleccionado. En mi caso, mis conocimientos de programación están centrados en el paradigma funcional,

principalmente. Sin embargo, el disponer de un lenguaje multiparadigma puede ser algo potencialmente beneficioso para el prototipo. Por ejemplo, si más adelante tuviera que ser actualizado, disponer de una primera versión desarrollada con un lenguaje multiparadigma permitiría que, si los futuros desarrolladores conocen más paradigmas que yo, los pudiesen explotar. En consecuencia, PHP y Python parecen mejores opciones.

Todos son gratuitos o se pueden encontrar alternativas gratuitas lo que no incrementa el coste de desarrollo del prototipo debido a la adquisición de licencias, etc.

Como se dijo en las subsecciones previas, el desarrollo de este primer prototipo no requiere del uso de bases de datos, pero esto no significa que en un futuro no pudieran ser de utilidad. Por ejemplo, si más adelante se busca que una misma aplicación web permita controlar varios pívots de riego, el uso de una base de datos sería de tremenda utilidad. Cada Raspberry Pi ya no enviaría la información de sus sensores y otros parámetros a una aplicación web intrínsecamente asociada a ella, sino que la enviaría a un servidor con una base de datos. Después, la aplicación web leería la información de todos sus pívots de esa base de datos. En consecuencia, esta característica no es necesaria en el momento actual pero dota a la solución final de un mayor potencial de crecimiento.

En relación a la dificultad de desarrollo de una aplicación web con cada uno de los lenguajes, el peor parado claramente es JSP. Al derivar de Java – un lenguaje complejo y difícil de aprender – JSP no es la mejor alternativa para proyectos relativamente pequeños, como sería nuestro caso. Por el contrario, esto mismo también le dota de otras características muy positivas como ser robusto o tener una buena gestión de errores.

Todos los lenguajes ofrecen un soporte en Internet más que suficiente, destacando los casos de PHP, Python y Ruby por ser lenguajes con un desarrollo y mantenimiento comunitarios. Del mismo modo, los cuatro disponen de plataformas de desarrollo que hacen la vida un poco más fácil a los programadores.

Considerando todos los aspectos puestos en discusión, considero que las mejores alternativas serían PHP y Python. JSP lo descarto dada la dificultad de aprendizaje y su modelo de paradigma exclusivamente orientado a objetos. A pesar de la mala adaptación de estas dos características al proyecto, sería una buena alternativa ya que cada día cuenta con más adaptación a dispositivos hardware genéricos como la Raspberry Pi, Arduino, etc [41], [42].

Ruby es de similares características a PHP y Python. Sin embargo, el ser un lenguaje desarrollado bajo una metodología y objetivo tan rompedor respecto de otros lenguajes así como su paradigma orientado a objetos solamente, hace que me decante por las otras dos opciones.

La mayor parte de la aplicación estará desarrollada en PHP dada su sencillez, rendimiento y prestaciones más que suficientes para el objetivo buscado. Sin embargo, también se utilizará de manera secundaria varios *scripts* en Python en lo relativo a la lectura de los sensores conectados a la Raspberry Pi. Esto no supone ningún problema pues ambos lenguajes son compatibles. En el código fuente en PHP se incluirían llamadas a funciones en Python cada vez que se quiera hacer una lectura de algunos de los sensores. Este enfoque simplifica mucho las cosas, pues con PHP sería más complicado y probablemente menos eficiente.

2.4 Otros elementos del software

En el capítulo de introducción, se estipuló que la aplicación web debía ser capaz de funcionar tanto en dispositivos fijos (PC) como móviles (smartphones, tabletas, etc.). Esta característica de valor añadido puede parecer trivial, pues cualquier código PHP correctamente programado funcionaría en cualquier dispositivo, pero en realidad no se refiere al mero hecho de ser capaz de interpretar el código en dispositivos móviles. Se refiere al hecho de que la aplicación web para el control remoto del pívot de riego debe de tener un grado de "usabilidad" (procedente del inglés, *usability*) en los dispositivos móviles equivalente al de los dispositivos fijos. La usabilidad es al software lo que la ergonomía sería a cualquier otro producto. Es decir, las condiciones de adaptación de un objeto a las características físicas y psicológicas de sus usuarios. Si una aplicación web es usable (o un producto tiene ergonomía), será más atractiva y de fácil utilización para sus usuarios. Si uno busca una definición un poco más formal, la Organización Internacional para la Estandarización (ISO, *International Organization for Standardization*) ofrece las dos siguientes:

- *"La usabilidad se refiere a la capacidad de un software de ser comprendido aprendido, usado y ser atractivo para el usuario, en condiciones específicas de uso"* [43].

- *"Usabilidad es la eficacia, eficiencia y satisfacción con la que un producto permite alcanzar objetivos específicos a usuarios específicos en un contexto de uso específico"* [44].

La primera definición hace referencia a las características internas y externas del software que contribuyen a su funcionalidad y eficacia. Sin embargo, dentro de la propia definición hay que destacar la parte en la que dice *en condiciones específicas de uso.* En otras palabras, la usabilidad no depende del producto únicamente, sino también del usuario y del entorno en el que se vaya a utilizar.

La segunda se centra en el concepto que hemos comentado al principio, el de capacidad de uso. Cómo un usuario es capaz de realizar una tarea específica en un escenario específico con efectividad. Volvemos a constatar de nuevo la dependencia con el usuario y el entorno.

En general, son cuatro los factores que determinan si un sitio web es "usable":

1) Tecnología de implementación adecuada (ya discutido en el subapartado anterior).
2) Contenido del sitio web: nuestro sitio web tendrá una función tan simple y específica que esto no se espera que vaya a constituir un elemento crítico.
3) Forma de presentación del contenido: este factor se abordará durante el propio desarrollo de la aplicación.
4) Interacción persona-ordenador: esta disciplina se considera el punto de convergencia de otros muchas como son la psicología aplicada, concretamente, la psicología cognitiva, la propia usabilidad web, la accesibilidad web, la ingeniería de diseño, la de desarrollo de hardware y software, etc. A lo largo del tiempo han ido surgiendo diversas metodologías que describen las técnicas para el diseño de aplicaciones con una IPO adecuada. La mayoría se basan en el hecho de que **los desarrolladores deben estimar lo más minuciosamente posible cuál va a ser el comportamiento del usuario.** Gracias a esta previsión del proceso cognitivo del usuario se conseguirá un resultado mucho más favorable. También hay otros modelos que proponen una retroalimentación constante entre usuarios y diseñadores con el fin del ir mejorando cada vez un poco más la aplicación. De entre estas metodologías, creo que la más conveniente para este proyecto es la de **diseño centrado en los usuarios** (UCD, *User Centered Design*). El diseño centrado en los usuarios es una moderna filosofía de

diseño en la que las preferencias y limitaciones de los usuarios del producto o aplicación final son el foco de atención en cada paso de proceso de diseño. La principal diferencia con otras filosofías de diseño es que esta trata de optimizar el diseño de la aplicación en torno a cómo los usuarios quieren o pueden utilizar la aplicación en lugar de obligar a los usuarios a cambiar su comportamiento para adaptarse a ella. Estas necesidades son consideradas tanto en las etapas del diseño como en el ciclo de vida de la aplicación en su totalidad.

Hasta no hace muchos años, dar una solución correcta a este problema hubiera supuesto tener que desarrollar varias aplicaciones, cada una diseñada de manera *ad-hoc* para un determinado dispositivo, que previamente se debería identificar. Afortunadamente, la explosión de los smartphones y tabletas no sólo benefició a los procesos de diseño del hardware sino también a los de software. Así, surgieron herramientas pensadas para facilitar la portabilidad del código entre dispositivos – desde una perspectiva a nivel de usuario final, no a bajo nivel (sistemas operativos, arquitecturas de máquinas, etc.). Se empleará una de estas herramientas para implementar esta característica de valor añadido.

2.4.1 jQuery Mobile

jQuery Mobile es un sistema de interfaz de usuario basado en HTML5 diseñado para hacer que los sitios webs y las aplicaciones se visualicen correctamente desde cualquier tipo de dispositivo. Es decir, es una interfaz de usuario que dota al software de capacidad de adaptación (término conocido en inglés como *responsive*). Esta idea queda muy bien recogida en su mantra "*write less, do more*" [45].

Con jQuery Mobile se puede diseñar un único sitio web de modo que funcione en cualquier tipo de smartphone, tableta o escritorio. La versión actual, gracias al uso de AJAX (*Asynchronous JavaScript And XML*) ofrece adaptabilidad completa a prácticamente la totalidad de dispositivos móviles y sus correspondientes sistemas operativos.

JQuery Mobile es un proyecto de la jQuery Foundation, es de código abierto y tiene un gran número de seguidores y desarrolladores por todo el mundo [45].

3

Resultados

Este capítulo tiene por objetivo presentar los principales resultados de este TFM para dar una visión global de los objetivos alcanzados y sus características.

3.1 Prototipo hardware

Como se explicó detalladamente en el capítulo anterior, este está basado en la Raspberry Pi modelo B y en la placa complementaria Custard Pi 2. Respecto al primer elemento, no hubo que realizar ninguna modificación o transformación a nivel de hardware sobre el mismo, pues se puede adquirir ya listo para su uso. Simplemente, se verificó que encendía/apagaba correctamente y detectaba la toma de alimentación. Por el contrario, en el caso de la placa Custard Pi 2, si fue necesario un proceso de soladura de todos los componentes y de verificación de la conexiones. Tras esto, se procedió al correcto acoplamiento de la placa Custard Pi 2 a la Raspberry Pi.

Finalmente, todo el prototipo hardware (Raspberry Pi + Custard Pi 2) se ajustó dentro de una caja protectora para evitar su deterioro por roces, fricciones o golpes. El resultado final puede verse en la Figura 8.

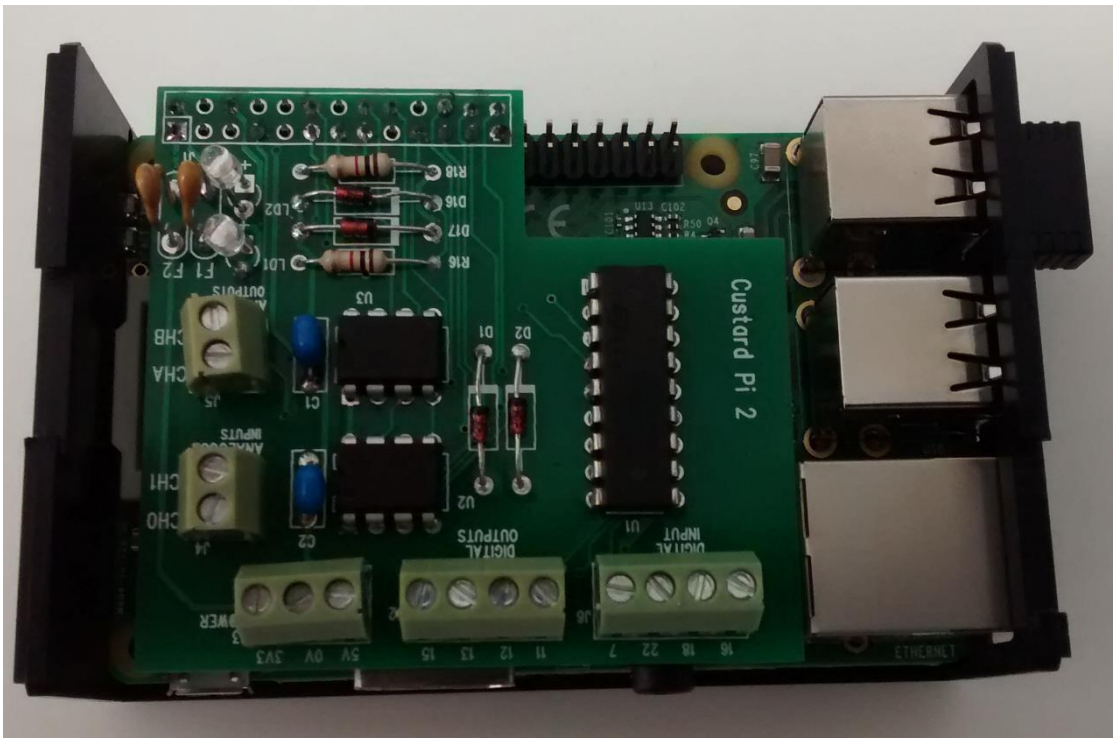


Figura 8. Imagen del prototipo hardware final. La caja protectora tiene otra mitad superior que no se muestra para poder ver el ajuste de Raspberry Pi + Custard Pi 2 a la misma. En la parte derecha, se puede observar el conector USB Wifi.

3.2 Prototipo software

Esta es la parte del trabajo en la que más decisiones de diseño hubo que tomar para alcanzar los objetivos y características planteados.

Si bien a la Raspberry Pi no fue necesario realizarle ninguna modificación o transformación a nivel de hardware, sí que fue necesario cargarle un sistema operativo y configurarle adecuadamente todas las conexiones (Wifi, Ethernet, etc.). Para ello, se hizo uso de una tarjeta de memoria SD en la que se realizaron dos particiones. En una partición se cargó el sistema operativo (Raspbian, [25]) y todo aquel software básico de funcionamiento (bibliotecas, controladores, etc.). Sobre esta partición, el usuario final del prototipo sólo tiene permisos de lectura, de modo que se establece una medida preventiva que evite el fallo del prototipo debido a una modificación o destrucción no controlada de su software base de funcionamiento. El tamaño de esta partición no superó el 30% de la capacidad de almacenamiento de la tarjeta de memoria SD.

La otra partición de la tarjeta de memoria SD se dedicó al almacenamiento del código del prototipo software así como otras aplicaciones o información que el usuario deseara almacenar. A continuación, se describirá el resultado alcanzado para dispositivos fijos y móviles así como las principales decisiones de diseño alcanzadas.

3.2.1 El prototipo software para dispositivos fijos (PC)

En aras de aumentar las posibilidades de éxito del prototipo se planteó la posibilidad de incluir un proceso de configuración inicial, de modo que el prototipo quedara asociado de manera personalizada a un determinado pívot de regadío según las preferencias del usuario. Un ejemplo de este proceso de configuración inicial puede verse en la Figura 9.

Este proceso de configuración inicial sólo se ejecuta la primera vez que el usuario final arranque el prototipo o cuando es reseteado. Durante el mismo, el usuario puede:

- Dar un nombre al pívot de regadío que será controlado por el prototipo.
- Determinar la longitud y la latitud de la posición en la que estará situado el pívot. Esta información será utilizada para el posicionamiento de los mapas, como se explicará posteriormente.

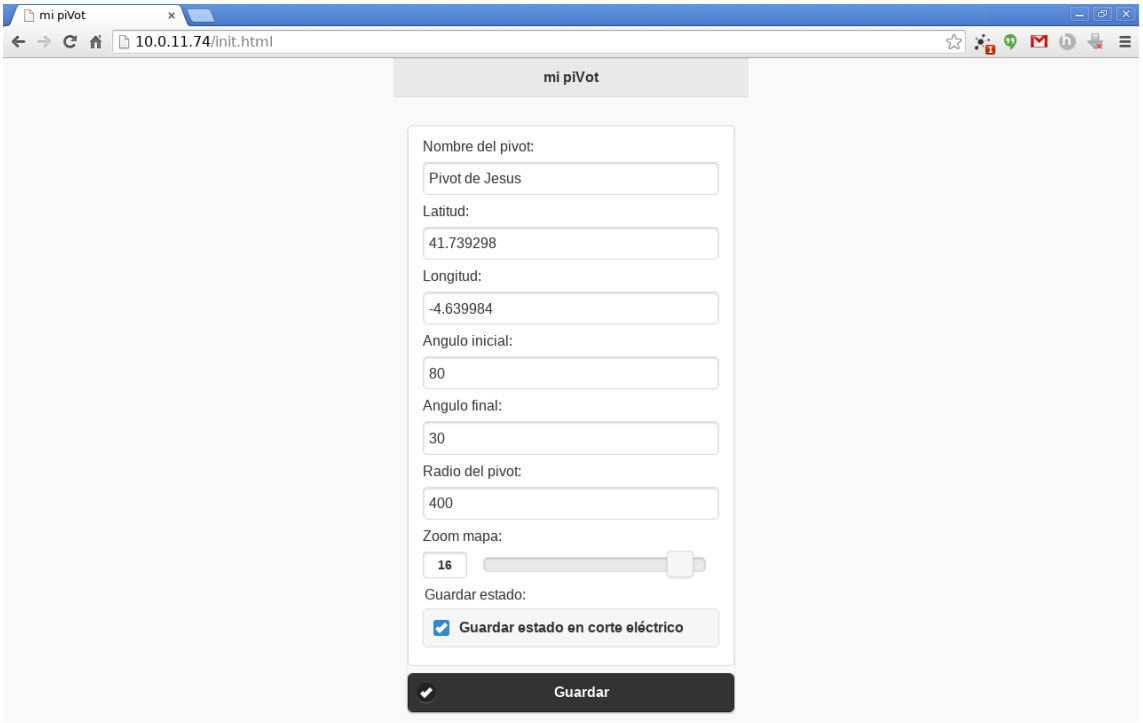


Figura 9. Pantalla de configuración inicial que el prototipo ofrece al usuario.

- Determinar el ángulo inicial y final de la superficie que regará el pívot. Aunque también existen pívots de avance lineal, los más comunes son los circulares. Estos, quedan caracterizados por la posición de su centro (latitud y longitud anteriores) y el ángulo que deben regar (respecto de su propio sistema de referencia). Mientras que el primer parámetro queda determinado por la posición de la toma de agua dentro de la finca y restringido en mayor o menor medida por la orografía de la misma – por ejemplo, si la toma de agua queda localizada en un pequeño cerro o colina es posible que sea necesario desplazarla a otro sitio ante la imposibilidad de salvar el desnivel – el segundo ofrece mucha mayor flexibilidad al usuario. Lo normal sería que todo pívot circular se emplease para regar el área que cubre en 360 grados a su alrededor. Sin embargo, esto no siempre es así por varias razones: (i) que el centro del pívot no coincida con el centro de simetría de la finca, en cuyo caso el riego sólo tendría sentido en un determinado ángulo; (ii) que, aunque el centro del pívot coincida con el de simetría de la finca, el usuario sólo desee regar una determinada región. Esta segunda opción puede tener sentido si, por ejemplo, toda la finca sufre un pequeño desnivel global. En este escenario, el tramo que queda en la parte superior debería ser regado más asiduamente pues, debido al desnivel, el agua tenderá naturalmente a la zona baja. Así, se facilita un control *ad-hoc* de la tecnología, lo que la hace más eficiente (se gasta menos agua) y más

efectiva (los cultivos de la zona baja no sufrirán de un exceso de agua, que en ocasiones es tan perjudicial como la sequía).

- Determinar el radio del pívot, es decir, la distancia desde el centro hasta el otro extremo. Esta información también será empleada por los mapas.
- Determinar el zoom inicial del mapa.
- Decidir qué hacer en caso de que se produzca un corte en el suministro eléctrico que afecta al pívot. Es decir, una vez que este se reanude, continuar el riego desde la última posición y/o configuración conocida o iniciar un nuevo proceso de riego desde cero.

Todos estos parámetros quedan automáticamente almacenados en un archivo de configuración (.cfg) que será leído cada vez que se arranque el pívot en lo sucesivo. Asimismo, el usuario podrá modificar el valor de algunos de estos parámetros (zoom del mapa) durante el funcionamiento del prototipo, aunque cada vez que lo apague y lo vuelva a encender tomará los valores del archivo de configuración.

Tras este primer paso de configuración, el usuario simplemente debe "logearse", como haría en cualquier servicio o aplicación web para poder acceder a los mapas y opciones de control del pívot. La página de *login* se muestra en la Figura 10. Consta de un mapa de fondo y una pequeña tabla en la parte superior derecha que permite introducir el nombre de usuario y la contraseña.

Si el usuario ha introducido correctamente su nombre de usuario y su contraseña, este accede a la página de control de su pívot, que se muestra en la Figura 11. Esta pantalla consta de dos partes claramente diferenciadas. La primera es el mapa y la segunda es la paleta de control de la esquina superior derecha.

La función del mapa no es otra que localizar el pívot en su posición geográfica real así como determinar el sector de riego del mismo. Para lograrlo se emplea como información base los parámetros de longitud, latitud y radio del archivo de configuración y la información del receptor GPS que el pívot lleva situado en su extremo móvil. Con los tres primeros parámetros podemos pintar sobre el mapa el "quesito" verde, mientras que con la información del receptor GPS podemos ir variando la posición de la línea roja en función del avance real del pívot. A medida que la línea roja avanza, en uno u otro sentido, el "quesito" va cambiando su color de fondo para destacar aquellas zonas que han sido regadas.

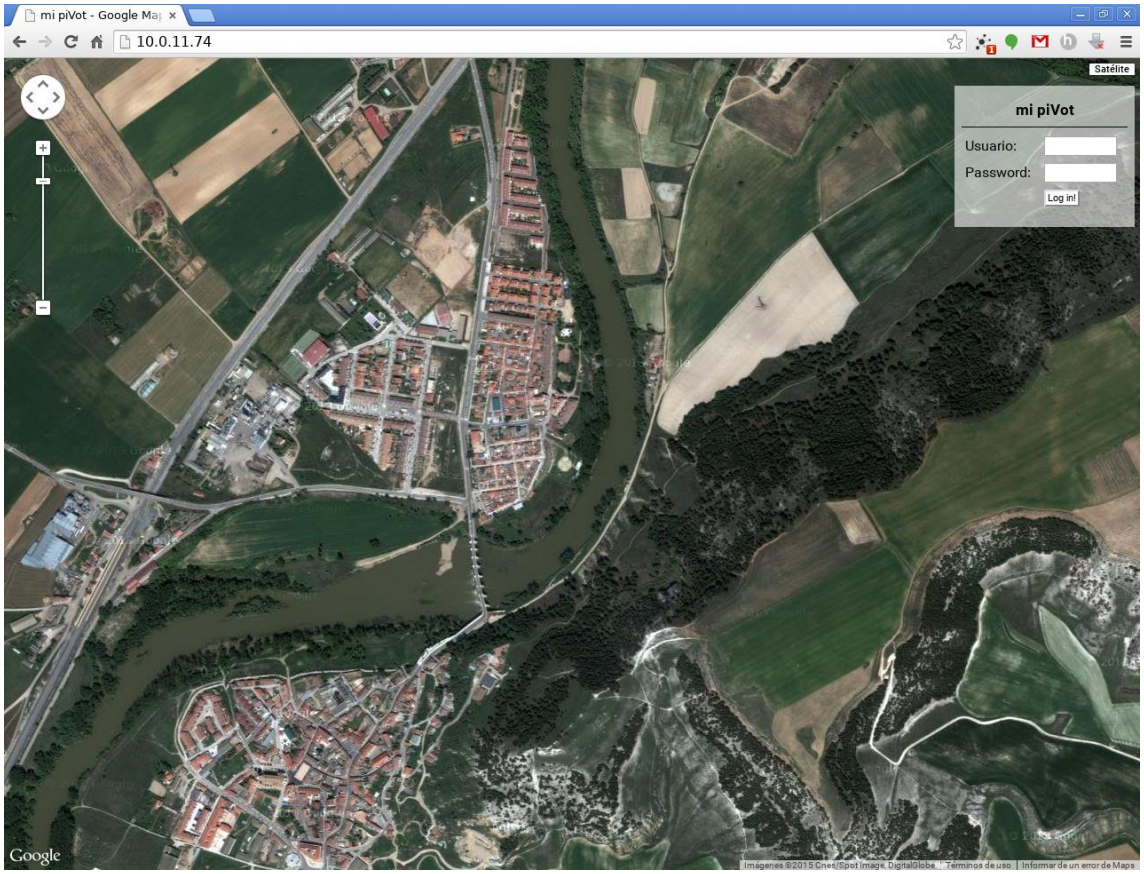


Figura 10. Pantalla en la que el usuario final debe "logearse".

El proceso descrito anteriormente, actualizar la posición de la línea roja y el color de fondo del "quesito", se ejecuta de manera permanente mientras el pívot de riego esté en régimen de trabajo. Con ello se evita que el usuario final deba tener abierta permanente la aplicación. Es decir, el usuario puede iniciar un proceso de riego desde su ordenador personal o dispositivo móvil, cerrar la aplicación y volver a acceder en cualquier momento futuro para simplemente monitorizar el avance del proceso o cambiar alguno de sus parámetros. La principal ventaja de este enfoque reside en que evita una transferencia de datos constante, y en general innecesaria, entre la Raspberry Pi el dispositivo del usuario final.

Aunque este proceso puede parecer computacionalmente costoso, se logró una implementación bastante eficiente que se describe a continuación. Una vez que el pívot de riego entra en proceso de funcionamiento, el prototipo software ejecuta de manera permanente cada tres segundos un bucle de control en el que se realizan las siguientes tareas:

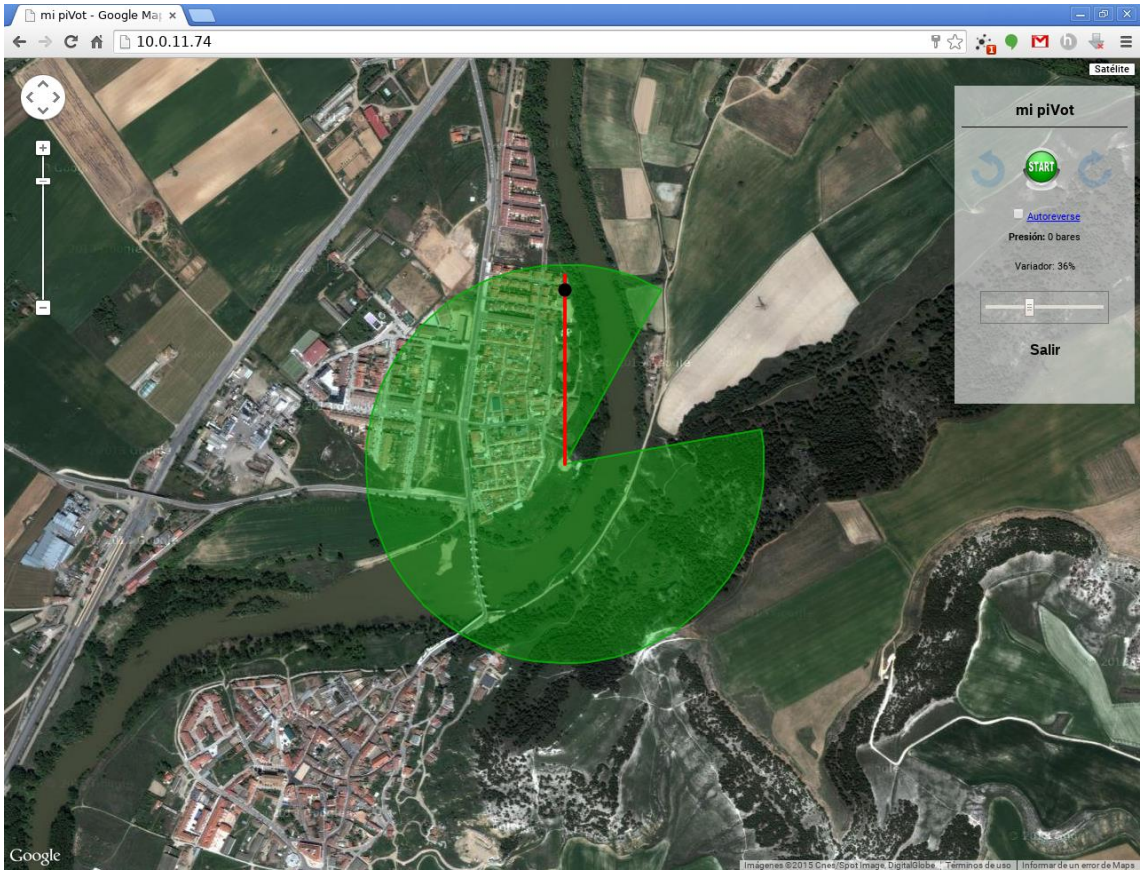


Figura 11. Pantalla de control del pívot.

1) Se configura un **array de estado actual**. Cada elemento de este array almacena el valor de alguno de los parámetros que describen el estado de funcionamiento del pívot, tanto los del panel de control como de otros secundarios que afectan a estos.

2) Se configura un **array diferencia**, como la comparación campo a campo del array de estado actual y el array de estado anterior. Este array diferencia es un array de valores booleanos. Es decir, valor lógico '1' si ha habido algún cambio en alguno de los parámetros (cambio provocado por el propio usuario por medio del panel de control o por el pívot en sí, por ejemplo, que haya terminado el proceso de riego) o valor lógico '0' en caso contrario.

3) En base al array diferencia, se llama a la **ejecución de una o varias de las funciones del prototipo software para el control del pívot.** Por ejemplo, la función que envía un correo electrónico al usuario informándole de que el proceso de riego ha finalizado o que ha habido una caída en la presión de la toma de agua o la función que obliga al pívot a cambiar su sentido de giro o simplemente detenerse.

4) El array de estado actual pasa a ser el **array de estado anterior** para la siguiente iteración del bucle de control.

Con este enfoque se obtienen las siguientes ventajas:

- Modularidad y escalabilidad del código del prototipo software: si en un futuro se quiere controlar algún otro posible evento, que quedará descrito por unos determinados valores y/o cambios en los parámetros del pívot, es tan sencillo como añadir una nueva función de control, lo que no afecta en absoluto a las funciones ya existentes.
- Tiempo de respuesta: gracias a la implementación anterior, no es necesario cargar en la memoria caché el código de todo el prototipo software, sino únicamente del proceso principal que incluye el bucle de control. A medida que vayan surgiendo nuevos eventos, se irá cargando el código de las funciones necesarias para dar una respuesta a cada uno de ellos.

Finalmente, para terminar la descripción de la sección del mapa, indicar que este se carga de manera local en la máquina del usuario final. En otras palabras, la descarga de toda la información relativa al mapa corre a cuenta de la conexión que este empleando el usuario final (conexión a Internet de su domicilio o la conexión 3G o 4G de su dispositivo móvil). De esta manera, prácticamente se reducen a cero las necesidades de descarga de datos del prototipo, lo que es positivo en términos de coste, tanto de operación como de mantenimiento.

Respecto a la paleta de control, en la Figura 11 se observa como está formada por diferentes elementos de acción y de información:

- Un botón *start and stop*. Su función es meramente la de arrancar o detener el pívot de riego.
- Botones de cambio de sentido del pívot: en caso del que el pívot esté en funcionamiento, se activan y permiten cambiar su sentido de giro.
- *Checkbox* para la función de auto-reverse: si la función auto-reverse está habilitada, cuando el pívot termina su ciclo de riego (un barrido completo al "quesito") lo que hace la Raspberry Pi es cancelar la orden de autómata de final de carrera que lleva incorporado naturalmente el pívot, lo que hace que vuelva a iniciar otro ciclo de riego en sentido contrario (en realidad no se cancela la orden del autómata de final de carrera, sino que esta se ejecuta con normalidad pero

seguidamente la Raspberry Pi vuelve a poner el pívot en funcionamiento). En caso de que el *checkbox* no esté activado, al finalizar un ciclo de riego el pívot se detiene.

- Seguidamente se muestra la presión de la toma de agua que alimenta al pívot. Este es un aspecto de su funcionamiento que no se puede controlar, ni mediante las Raspberry Pi ni mediante ninguno de los elementos propios del pívot. Sin embargo, sí parece interesante informar al usuario final del valor que toma este parámetro pues es muy posible que, de manera heurística, sepa cómo afecta un valor u otro al rendimiento del pívot. De la misma forma, cuando su valor caiga fuera de los límites marcados por el fabricante del pívot, se enviará automáticamente un correo electrónico al usuario final para informarle de tal situación.
- El variador: este parámetro controla el régimen de trabajo del pívot. Su valor se controla mediante la barra situada debajo del mismo. Básicamente, gracias a este elemento de acción se permite al usuario final determinar qué cantidad de agua quiere emplear en su riego. Si por ejemplo marca 70%, en un minuto el pívot avanzaría durante 42 segundos y estaría detenido durante otros 18. Siempre arroja agua, por lo que durante el tiempo de detención inunda ligeramente la zona. Cuando se mueve, su velocidad de avance es siempre la misma. Así, pueden existir cultivos que necesiten una frecuencia de riego mayor pero no por ello mayor cantidad de agua, mientras que otros lo contrario. En este mismo sentido, gracias al variador se puede alcanzar un mayor acompasamiento entre los ciclos de riego y las lluvias naturales. Por ejemplo, si justo antes de que a una determinada zona se le autorice el riego ha habido una jornada de lluvias, es muy probable que no sea necesario sobrecargar a la planta con tanta agua.

En último lugar, indicar que se ha preferido no extender más la descripción del funcionamiento del prototipo para no recargar excesivamente esta memoria, pues es una tarea ardua tanto en la redacción como en la interpretación.

3.2.2 El prototipo software para dispositivos móviles

En este apartado se describirá el prototipo de software que se empleará en el caso de dispositivos móviles como smartphones, tabletas, etc. La descripción será meramente

estética, pues las funcionalidades que ofrece son exactamente las mismas que para los dispositivos fijos.

En la Figura 12 puede verse la pantalla de configuración inicial del pívot en el caso de dispositivos móviles. Su presentación es la misma que en el caso de dispositivos fijos ya que los campos a introducir y los botones de acción (*checkbox*, la barra del variador) se pueden manejar indistintamente con cualquier tipo de dispositivo. Lo que se consigue con jQuery Mobile es que el tamaño de los campos y botones de acción se adapte de manera automática a las prestaciones (tamaño de pantalla, presencia de teclado o no) de cada dispositivo móvil.

En la Figura 13 se recoge la pantalla de acceso a la zona de control del pívot. Finalmente, en la Figura 14 se muestra la pantalla de control. En esta ocasión existe una diferencia fundamental respecto al caso de dispositivos fijos, se ha suprimido el mapa. Teniendo en cuenta el reducido tamaño de estos dispositivos – si bien en esta característica existe mucha variabilidad según modelos, marcas, etc. – no parece lo más adecuado sobrecargar un espacio reducido incluyendo tanto el mapa como el panel de control. Esta decisión de diseño queda apoyada por las siguientes razones:

1) La función del mapa en el prototipo software para dispositivos fijos es meramente complementaria. Es decir, simplemente facilita el control del régimen de trabajo del pívot en un único golpe de vista, pero no ofrece ninguna funcionalidad.

2) Anteriormente se comentó que la carga del mapa corre a cuenta de la conexión local del usuario final. En el caso de dispositivos móviles, esta conexión se suele basar en las redes 3G o 4G, en las que el usuario tiene una cantidad limitada de datos a descargar. En base a lo expuesto en el punto primero, la eliminación del mapa es aquello que permite contrarrestar esta restricción previsible del usuario final sin mermar considerablemente la calidad o funcionalidad de todo el prototipo.

3) La ausencia del mapa no supone en ningún momento que el usuario final no tenga capacidad de monitorización del régimen de trabajo del pívot, pues siempre que ocurriera algún evento que alterara su rendimiento se lanzaría una alerta por medio de un correo electrónico (tanto en dispositivos fijos como móviles).

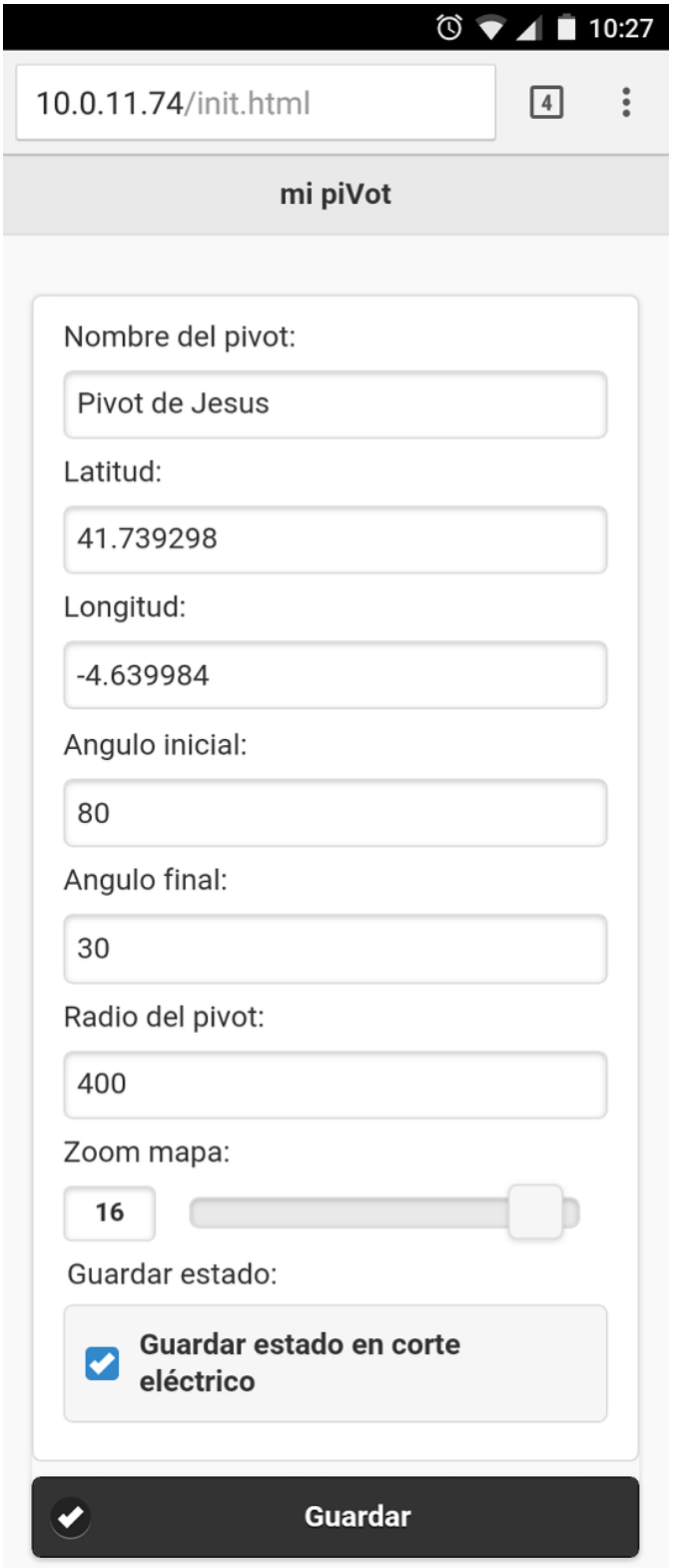


Figura 12. Pantalla de configuración inicial en el caso de dispositivos móviles.

4) Finalmente, eliminar el mapa aumenta la *usabilidad* de la aplicación web en los dispositivos móviles. Es decir, el usuario final trabajará de manera mucho más cómoda con una paleta de control que ocupe toda la pantalla de su dispositivo

en vez de únicamente un 15% o un 20%, que es lo que ocuparía en caso de estar presente el mapa.

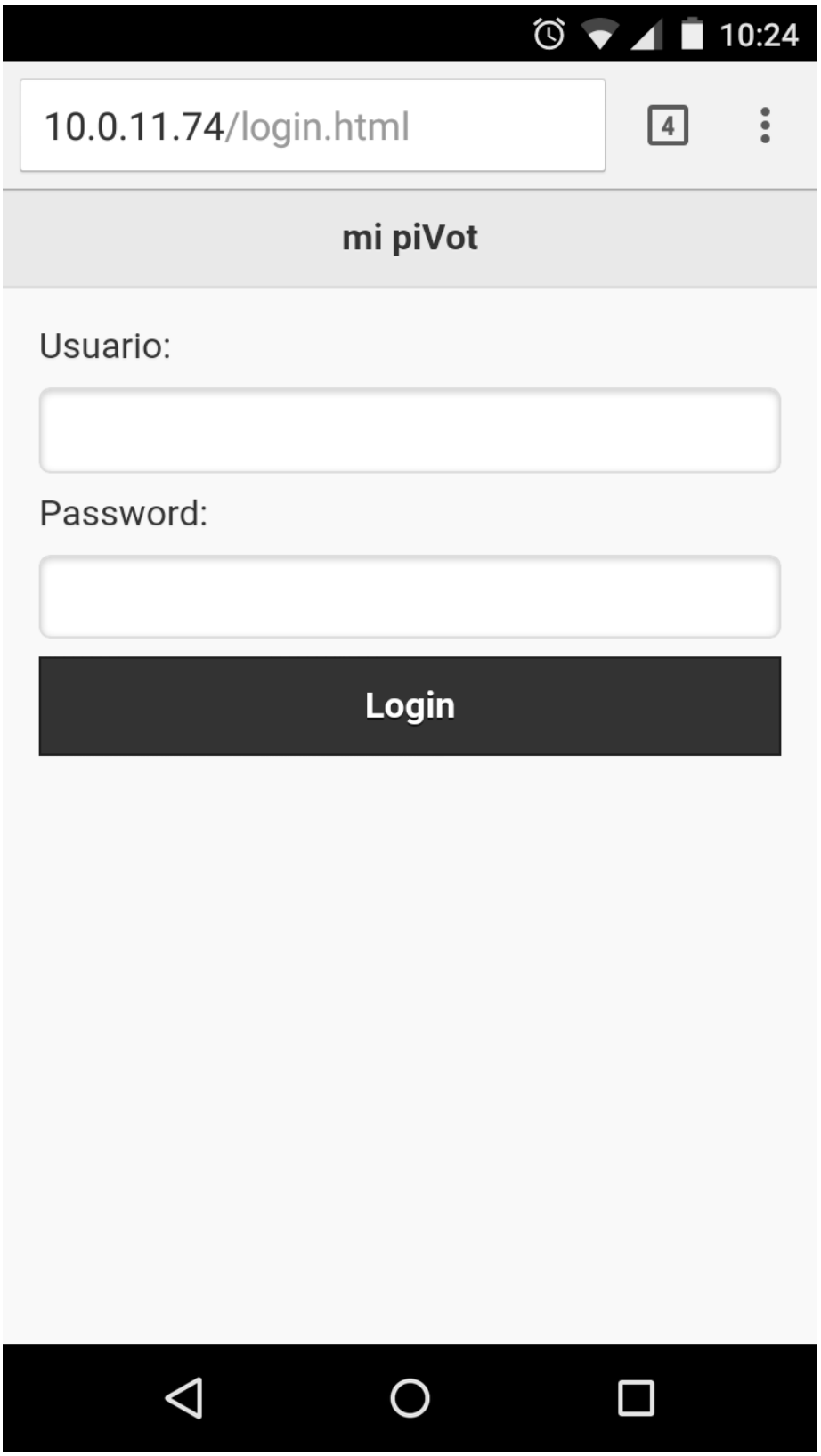


Figura 13. Pantalla de *login* en el caso de dispositivos móviles.

Figura 14. Panel control para el caso de dispositivos móviles.

3.3 Análisis técnico-económico de operadoras móviles

3.3.1 Introducción

Llegados a este punto, para poder considerar como logrado el objetivo principal de este trabajo se requiere que el prototipo disponga de algún tipo de conexión que permita al usuario final un control a distancia del mismo.

Se han considerado dos alternativas para dotar de conectividad a nuestro prototipo:

1) Wifi: sin duda, sería la opción más económica de todas, pues nos beneficiaríamos de la velocidad y prestaciones de una conexión por cable que posteriormente se transforma en una red inalámbrica. Sin embargo, considerando el entorno de operación, esta elección parce poco viable. Aunque en la actualidad se puede lograr la extensión de una red Wifi a largas distancias (km) mediante el uso de repetidores, esta alternativa encarecería enormemente el coste de operación y mantenimiento del sistema. Además, su instalación requiere de unos conocimientos técnicos algo más avanzados que los esperados para el usuario final. Únicamente en el caso de regadíos situados muy próximos a poblaciones o quizá explotaciones ganaderas tendría sentido esta opción. En cualquier caso, siempre se puede ofrecer el prototipo con el conector USB-Wifi, Figura 8, incorporado (su coste es irrisorio, menos de un euro) para que en caso de que sea posible, esta opción se convierta en realidad.

2) Conectividad por datos 3G/4G: la evolución de la telefonía móvil ha hecho que, actualmente, las operadoras ya no sólo se limiten a ofrecer servicios de voz o mensajes de texto sino que al cliente se le ofrece paquetes que incluyen además de estos servicios conectividad mediante 3G o 4G, aunque generalmente con una limitación en la cantidad de datos a descargar. De la misma forma, el éxito comercial de estos servicios de telecomunicaciones ha llevado a las principales operadoras a mejorar cada vez más la calidad y extensión de sus áreas de cobertura, de lo que se han beneficiado, sobre todo, las zonas rurales. Este hecho resulta enormemente positivo para este proyecto pues las tierras de regadío se sitúan en torno a las cuencas fluviales y, por lo tanto, relativamente próximas a los núcleos de población rural o vías de comunicación (carreteras, autovías, autopistas, líneas férreas de alta velocidad, etc.). De este modo, si estos núcleos

de población o vías de comunicación tienen cobertura, nosotros nos podremos beneficiar de ella para dotar de conectividad a nuestro prototipo.

Por todo esto, el objetivo de este subapartado será analizar y comparar varias ofertas de distintas compañías de telefonía móvil que pudieran constituir una alternativa a la hora de ofrecer conectividad a nuestro prototipo. Dada la variabilidad temporal de este tipo de ofertas, no se espera de este subapartado un análisis profundo de todo el mercado, sino simplemente ofrecer un punto de referencia para una eventual transferencia del prototipo al mercado y escenarios de trabajo reales.

3.3.2 Análisis de la extensión y calidad de la cobertura móvil en la Comunidad Autónoma de Castilla y León

En España existen cuatro operadoras de móvil con red propia: Movistar, Vodafone, Orange y Yoigo. Además, existen los denominadas Operadores Móviles Virtuales (OMV) que son compañías que utilizan la red de alguna de las cuatro anteriores para prestar sus servicios, es decir, no poseen una concesión de un rango del espectro de frecuencias y por lo tanto no tienen una infraestructura propia de antenas. Las OMV se caracterizan por ofrecer unas tarifas muy bajas – ya que no tienden a subvencionar terminales de la misma forma que las grandes para captar clientes – y una mayor flexibilidad al usuario a la hora de elegir los servicios contratados [46].

Al contrario de lo que piensa mucha gente, la OMV tienen exactamente la misma cobertura que aquella compañía con red propia a la que han alquilado su infraestructura. Por lo tanto, para analizar la extensión del mapa de cobertura en España hemos de analizar la cobertura de las cuatro compañías con red propia [46]. Posteriormente, una vez sepamos qué compañía tiene un grado de cobertura mayor, se analizará qué OMV es la que mejor se adapta al servicio buscado. En este sentido es importante resaltar que, en numerosas ocasiones, las OMV son creadas por empresas mayores – casos más frecuentes: banca, cadenas de supermercados u otra operadora móvil con red propia – para cubrir un determinado servicio y/o grupo específico de clientes [47]. Por ejemplo, Alów es una OMV enfocada a la comunidad de extranjeros en España (inmigrantes, turistas, estudiantes extranjeros) de ahí que su mayor valor sean unas tarifas de llamada al extranjero muy asequibles (0.14 cént/min a EEUU, Canadá, Rusia, etc.); El grupo bancario Bankinter creó una OMV del mismo nombre con tarifas tanto de voz como de

voz más datos para ofrecer alertas y aplicaciones bancarias gratuitas a sus clientes; La cadena de supermercados Eroski creó Eroski Móvil a fin de fidelizar a sus clientes, de modo que se pueden conseguir minutos gratis mediante compras en el supermercado. Otras OMV están enfocadas a sectores profesionales concretos como autónomos (Ibercom/Xtra Telecom) o a los adolescentes (Tuenti Movil). Finalmente, existen OMV que se caracterizan por ofrecer una personalización en las tarifas muy grande – sin permanencia, "pay on go", etc – por ejemplo, Lowi [47]. A priori, estas últimas serán las de mayor interés para el presente proyecto.

Análisis de la cobertura 2G, 3G y 4G de las operadoras con red propia:

A fin de simplificar un poco el análisis y comparativa, nos centraremos únicamente en la Comunidad Autónoma de Castilla y León.

La compañía Orange ofrece cobertura 4G en todas las ciudades capital de provincia y otros núcleos de población importantes como Aranda de Duero, Medina del Campo, Ponferrada, etc. Sin embargo, todavía sigue existiendo una amplia parte del territorio fuera de la cobertura 4G, por lo que parece necesario analizar la cobertura 3G. En este caso, existe una mayor zona cubierta, aunque sigue habiendo amplias zonas fuera de ella. Finalmente, la cobertura 2G es prácticamente total, con la excepción de ciertas zonas de sombra en núcleos montañosos o de difícil acceso. En la Figura 15 se muestran los tres mapas de cobertura descritos [48].

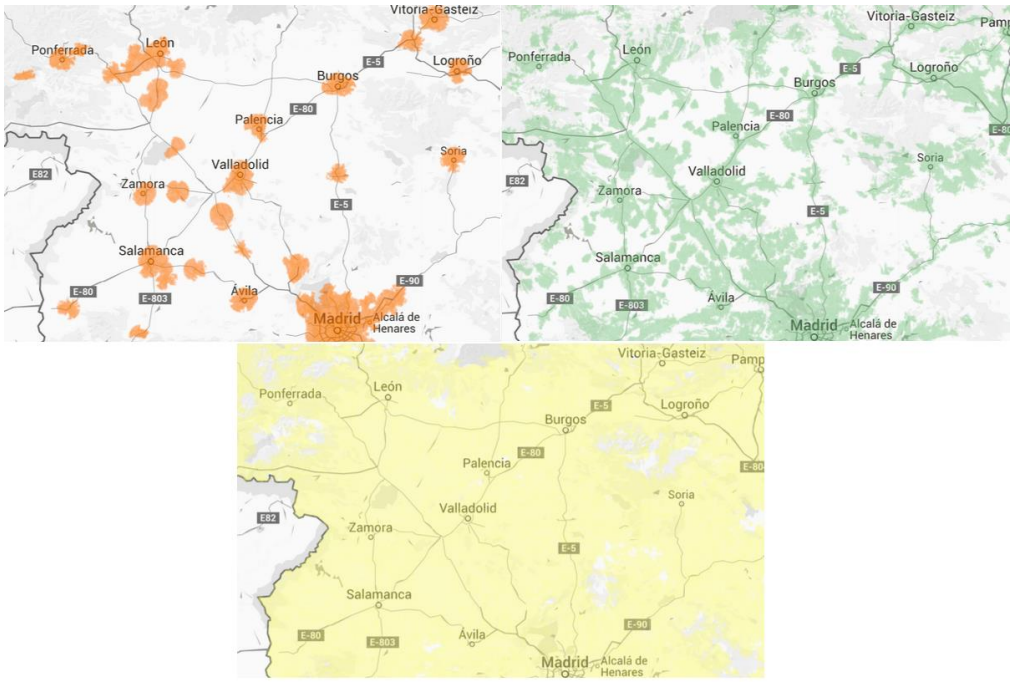


Figura 15. Mapas de cobertura de la compañía Orange: 4G (naranja), 3G (verde) y 2G (amarillo).

La cobertura 4G de la compañía Vodafone es similar a la de Orange. Sin embargo, en base a la Figura 16, su cobertura 3G parece tener un mayor despliegue. Finalmente, la cobertura 2G se podría considerar equivalente al caso anterior [49].

En el caso de la compañía Movistar, se produce un ligero contraste con Orange y Vodafone. La cobertura 4G se concentra en las capitales de provincia casi exclusivamente pues existen pocas poblaciones secundarias que puedan disfrutar de ella. Respecto a la cobertura 3G, se puede considerar bastante similar a la de Vodafone. Por el contrario, la cobertura 2G es prácticamente total, existiendo un nivel alto de la misma en prácticamente toda la comunidad autónoma. Sus mapas de cobertura se muestran en la Figura 17 [50].

En último lugar, se analizó la compañía Yoigo. La cobertura 4G de esta compañía se centra exclusivamente en las capitales de provincia de nuestra Comunidad Autónoma, de manera similar a Movistar. Su cobertura 3G es similar a la de Vodafone o Movistar mientras que su cobertura 2G se puede considerar prácticamente total. Sus mapas de cobertura se muestran en la Figura 18 [51].

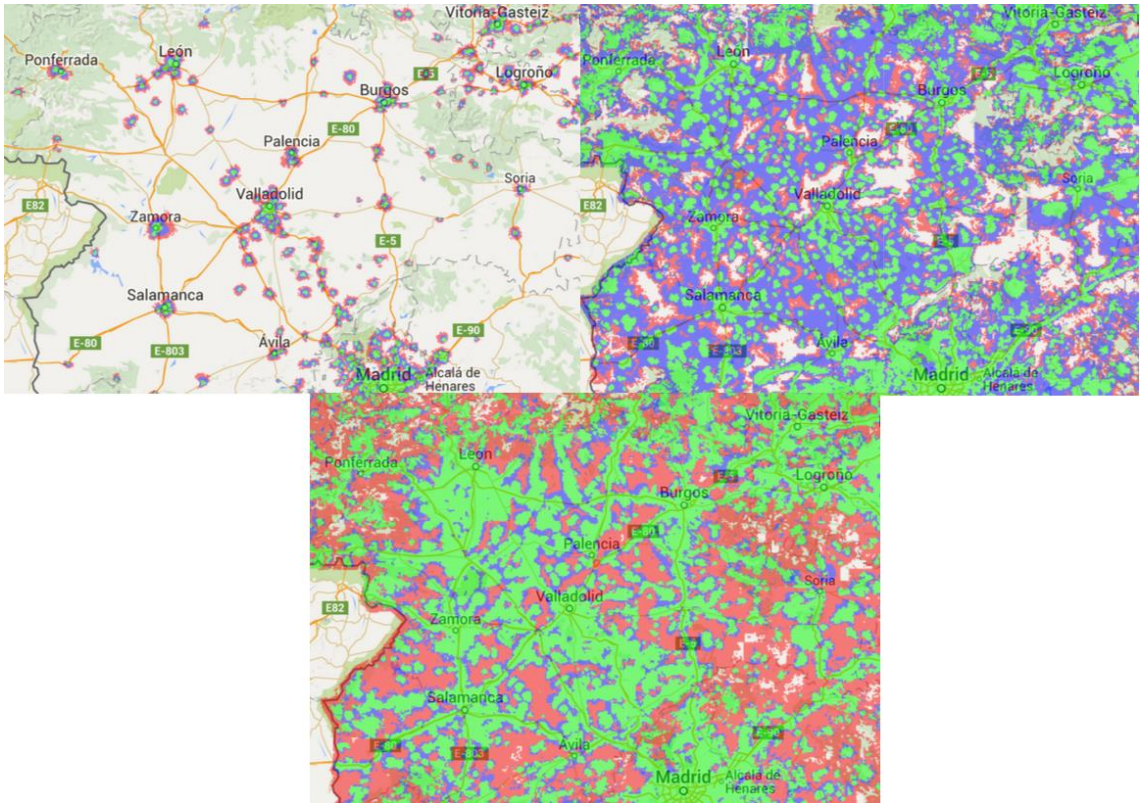


Figura 16. Mapas de cobertura de la compañía Vodafone: 4G (arriba izq.), 3G (arriba dcha.) y 2G (abajo centro). Leyenda de colores: verde, muy alta; azul, alta; rojo, baja.

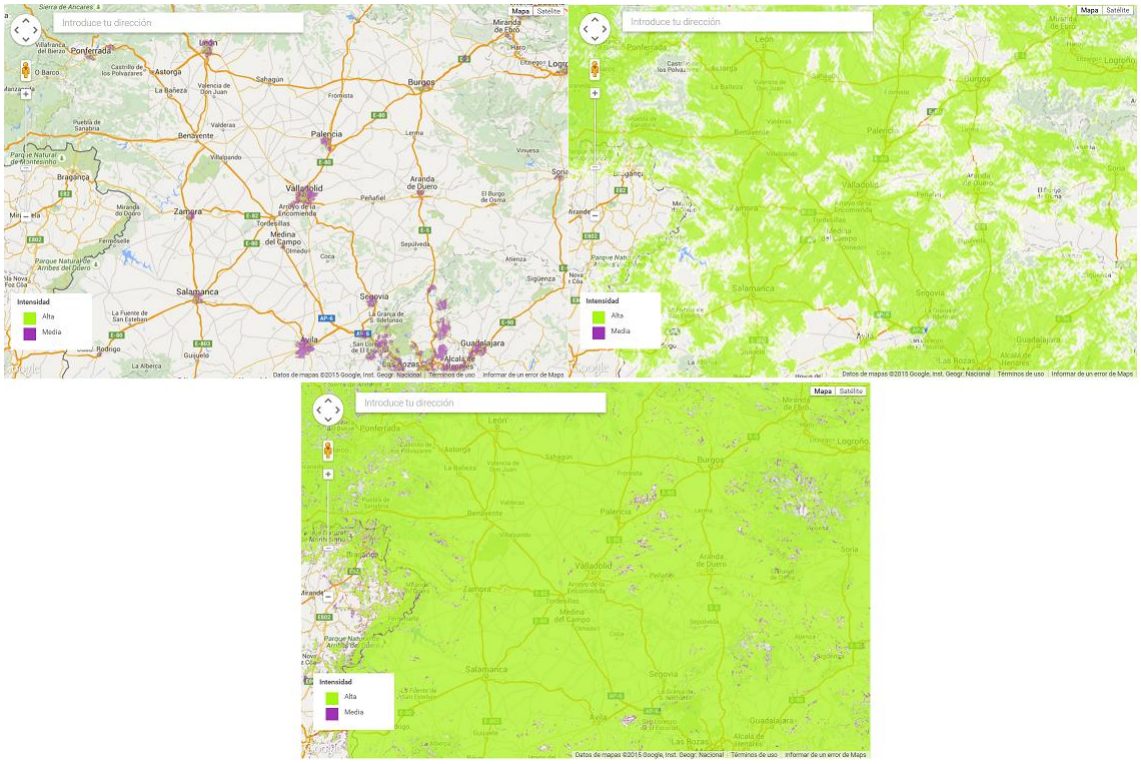


Figura 17. Mapas de cobertura de la compañía Movistar: 4G (arriba izq.), 3G (arriba dcha.) y 2G (abajo centro). Leyenda de colores: verdad, alta; morado, media.

Se puede concluir por lo tanto que las cuatro compañías de telefonía móvil con red propia en España ofrecen un grado de cobertura similar en redes 2G, 3G y 4G. Orange y Vodafone destacan por una cobertura 4G extendida a poblaciones secundarias. La cobertura 3G de Vodafone aparenta ser ligeramente superior a las del resto, mientras que la de Orange un poco inferior. En relación al 2G, Yoigo y Movistar son las que ofrecen la máxima extensión de cobertura.

El hecho de que las cuatro ofrezcan una cobertura similar es en parte positivo, en parte negativo para el proyecto. Es en parte positivo porque, de este modo, disponemos de una mayor oferta de OMV con las que poder contratar el servicio buscado. Por el contrario, si una de ellas hubiera tenido un grado de cobertura muy superior a las otras tendríamos solucionado el problema de la conectividad para nuestro prototipo, pero al estar sujetos a la red de una determinada compañía, el número de OMV sería menor, lo que podría llevar a que no encontráramos un operador que se adapte a los servicios que buscamos.

Finalmente, siempre que se realiza un análisis de este tipo conviene contrastar la información de las fuentes originales con alguna otra fuente externa. Existen en Internet numerosas páginas web y aplicaciones que ofrecen análisis alternativos.

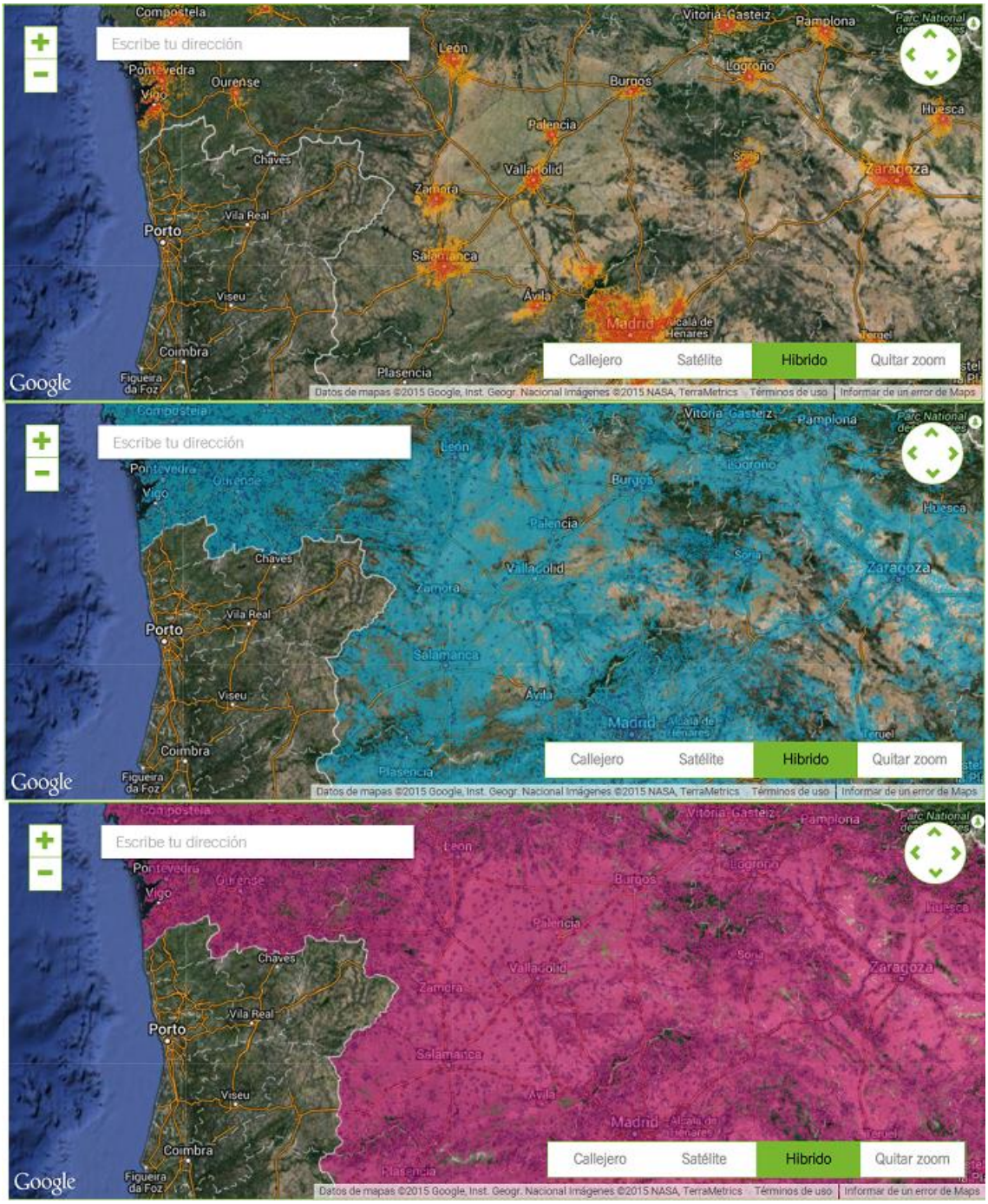


Figura 18. Mapas de cobertura de la compañía Yoigo: 4G (arriba), 3G (centro) y 2G (abajo).

Una de las más conocidas es la página web OpenSignal [52], donde se puede analizar y comparar la cobertura móvil (2G, 3G y 4G) de numerosos países. Para el caso de Castilla y Léon, esta página determina que la mejor cobertura 4G es la de Vodafone, seguida de cerca por la de Orange, y un poco más lejos Movistar y Yoigo (véase el resto de datos en la Tabla 4). La mejor cobertura 3G nuevamente es ofrecida por Vodafone, seguida de Yoigo, Orange y Movistar, si bien en este caso y, en base a los datos de la Tabla 5, la diferencia entre ellas es menor.

Operadora	Descarga [Mbps]	Subida [Mbps]	Ping [ms]	Fiabilidad [%]
Vodafone	23.59	10.68	58	97.15
Orange	17.96	7.25	68	97.00
Movistar	13.53	5.15	69	97.35
Yoigo	13.13	5.76	75	96.61

Tabla 4. Comparativa por operadoras de las características de las redes 4G en Castilla y León.

Operadora	Descarga [Mbps]	Subida [Mbps]	Ping [ms]	Fiabilidad [%]
Vodafone	5.07	0.88	96	91.98
Yoigo	5.28	1.10	120	93.57
Orange	3.97	0.84	113	91.31
Movistar	4.88	0.56	109	92.07

Tabla 5. Comparativa por operadoras de las características de las redes 3G en Castilla y León.

Curiosamente, y un poco en contradicción con los datos oficiales, la mejor cobertura 2G es la de Vodafone, sobre todo en lo que a capacidad de descarga se refiere. Según los datos de la Tabla 6, le siguen Yoigo, Orange y Movistar. OpenSignal también ofrece la posibilidad de realizar comparativas combinadas de varias redes. Por ejemplo, 2G y 3G o 3G y 4G. Para ambos casos, la clasificación de operadoras es la misma que cuando se compara únicamente la red 3G (Tabla 7 y Tabla 8). En último lugar, si hacemos una comparativa de todas las redes, Vodafone sigue siendo la mejor y Movistar la peor, pero Orange asciende al segundo puesto y Yoigo cae al tercero (Tabla 9).

Operadora	Descarga [Mbps]	Subida [Mbps]	Ping [ms]	Fiabilidad [%]
Vodafone	0.97	0.04	568	61.34
Yoigo	0.22	0.04	486	66.96
Orange	0.13	0.04	528	69.76
Movistar	0.26	0.02	458	69.91

Tabla 6. Comparativa por operadoras de las características de las redes 2G en Castilla y León.

Operadora	Descarga [Mbps]	Subida [Mbps]	Ping [ms]	Fiabilidad [%]
Vodafone	5.01	0.86	116	90.59
Yoigo	5.01	1.00	155	90.74
Orange	3.86	0.81	135	90.03
Movistar	4.75	0.52	140	89.96

Tabla 7. Comparativa combinada por operadoras de las características de las redes 2G y 3G en Castilla y León.

Operadora	Descarga [Mbps]	Subida [Mbps]	Ping [ms]	Fiabilidad [%]
Vodafone	10.40	3.70	85	93.45
Yoigo	7.54	2.44	108	94.44
Orange	7.78	2.52	101	92.83
Movistar	6.85	1.62	100	93.22

Tabla 8. Comparativa combinada por operadoras de las características de las redes 3G y 4G en Castilla y León.

Operadora	Descarga [Mbps]	Subida [Mbps]	Ping [ms]	Fiabilidad [%]
Vodafone	10.26	3.48	94	91.93
Orange	7.53	2.35	121	90.48
Yoigo	6.96	2.38	144	91.41
Movistar	6.68	1.55	126	91.33

Tabla 9. Comparativa combinada por operadoras de las características de las redes 2G, 3G y 4G en Castilla y León.

Los datos cuantitativos ofrecidos por la web OpenSignal [52] vienen a confirmar en gran medida los resultados que se habían obtenido a partir de un análisis cualitativo de las fuentes oficiales. La conclusión que se extrae es que la mejor operadora en Castilla y León es Vodafone y la peor Movistar. Orange constituye el punto de equilibrio, pues no es nunca ni la mejor ni la peor mientras que Yoigo ofrece buena cobertura 2G y 3G pero en cobertura 4G es peor que sus rivales.

3.3.3 Comparativa de la oferta de servicios de diferentes operadoras móviles

Teniendo en cuenta toda la información anteriormente analizada, en este subapartado se comparará la oferta de servicios de diferentes operadores móviles. Nuestra operadora ideal sería aquella que ofrecería un servicio de datos 3G o 4G sin contrato de permanencia y al menor precio posible. Respecto al límite de descarga de datos, no se espera que esto sea una limitación importante pues el prototipo ha sido diseñado para que exista la menor transferencia posible de datos entre él y el usuario final. En cualquier caso, también se tendrá en consideración este aspecto.

Debido al gran número de operadoras existentes, sólo se incluirán en esta comparativa aquellas que se consideren más relevantes. Otras, ni siquiera se compararán pues como se explicó antes, están enfocadas a usuarios completamente distintos a nosotros.

En Internet existen multitud de webs que permiten comparar tarifas de distintas compañías. Algunas de las más destacadas son: Comparaiso [53], Kelisto [54] o Rastreator [55], etc. Si introducimos los parámetros anteriormente mencionados, las compañías seleccionadas son las siguientes:

- **Simyo:** ofrece tarifas tanto de contrato como de prepago pero todas sin cuota inicial o final y sin contrato de permanencia. Las tarifas de prepago las puede configurar el usuario según sus gustos por medio de los denominados bonos.

Existen bonos de datos, de minutos y de minutos a simyos. También existen paquetes que combinan un determinado bono de datos más otros de minutos. En caso de que el usuario no elija ningún bono, se le aplica la tarifa sin cuotas, en la que si no hay ningún consumo, no hay ningún cobro. Los bonos de datos son de 100 MB, 300 MB, 1GB, 2GB, 4GB, y sus respectivos precios, 1, 2.5, 6, 11, y 21 €/mes. Si el usuario consume su bono antes de final de mes, pasará a la tarifa sin cuotas (3.63 cét/MB IVA incluido) o podrá contratar otro bono. Los bonos se descuentan del saldo que se haya introducido en la tarjeta. La cobertura de estos bonos de datos es 3G. El usuario puede añadir, cambiar o quitar los bonos desde el área personal de la web de Simyo o llamando al 121. Simyo es una OMV que pertenece a la compañía holandesa KPN pero que en España la adquirió Orange, por lo que opera con su red, y constituye junto con Amena la segunda *low cost* de esta gran operadora telefónica [56].

- **Lowi:** esta compañía ofrece una tarifa de contrato, sin permanencia ni cuota inicial o final por 6 €/mes. La tarifa incluye 1GB de datos y llamadas gratis (sólo se cobra el establecimiento de llamada: 18.15 céntimos de euro). Si un mes el usuario no consume todos sus datos, la diferencia se acumula para el mes siguiente. Por el contario, si se supera el consumo de datos contratado, la velocidad se reduce a 32 Kbps. Asimismo, el usuario puede añadir cómodamente más MB a su tarifa. Por ejemplo, 200 MB por 1 €, 400 MB por 2 €, etc. Lowi es la OMV lanzada por Vodafone para estar presente en el mercado *low cost* [57].
- **MásMóvil:** al igual que Simyo, esta compañía permite al cliente configurar tarifas (voz y datos) tanto de contrato como de prepago. También dispone de los denominados bonos de datos para tabletas o PC, tanto en formato prepago como contrato. No exige permanencia ni cuota inicial o final. Para nuestro caso, serían los bonos de datos prepago el servicio que mejor se adaptaría a lo buscado. Ofrece bonos de 300 MB, 600 MB, 1GB, 2 GB, 5 GB y 10 GB a los siguientes respectivos precios: 3, 6, 9, 16, 40 y 75 €/mes. MásMóvil es una OMV que opera sobre la red de Orange [58].
- **PepePhone:** esta OMV ofrece tanto servicios móviles como ADSL. Como las anteriores, no exige permanencia ni pago inicial o final. Sólo dispone de tarifas que combinan voz y datos. La más básica incluye un 1GB de datos y llamadas gratis si duran menos de 20 minutos (después se cobra a 0.6 céntimos de € el

minuto; también se cobra el establecimiento de llamada a 18.15 céntimos de €). PepePhone es una OMV nacida en el seno del grupo turístico español Globalia que opera con la red de Movistar [59].

- **Movizelia:** ofrece una única tarifa que se ajusta totalmente al consumo del usuario tanto en llamadas como en datos. Las llamadas se cobran a 8.35 céntimos de € el minuto y los datos móviles a 2.3 céntimos de € el Mb. No se cobra por el establecimiento de llamada y se puede llamar a más de 70 destinos internacionales. No exige compromiso de permanencia pero sí una cuota mensual de 2.3 €. Movizelia es una OMV que opera bajo cobertura de Orange [60].
- **Amena:** la otra OMV de Orange ofrece una tarifa de contrato por 6.95 €/mes con un 1GB de datos y llamadas gratuitas (sólo se cobra el establecimiento de llamada). No tiene compromiso de permanencia ni cuotas iniciales o finales. La cobertura de los datos es mediante la red 4G de Orange [61].

El resto de compañías analizadas, OMV u operadoras con licencia propia, ofrecían una serie de tarifas alejadas del servicio buscado o con un precio excesivo, por lo que no han sido incluidas en el análisis. Todos los precios que se han expuesto son IVA incluido.

Se han analizado seis OMV que podrían ser una buena alternativa para dar conectividad a nuestro prototipo. Cuatro de ellas – Simyo, MásMóvil, Movizelia y Amena – operan con la red Orange, PepePhone con la red de Movistar y Lowi con la red de Vodafone. Destacar la flexibilidad de Simyo y MásMóvil gracias a su formato de bonos de datos. Lowi ofrece una tarifa a un precio muy llamativo, sin embargo, está más enfocada a un usuario típico del servicio de telefonía y datos móviles. Lo mismo les ocurre a PepePhone y Amena. Finalmente, Movizelia ofrece un servicio "pay on go" a un precio asequible.

A priori, Simyo y MásMóvil serían las dos mejores operadoras para dotar de conectividad a nuestro prototipo. Además, ambas operan con la red Orange, que en la clasificación global anteriormente expuesta estaba en segunda posición como operadora con mejor cobertura de red en Castilla y León. Los paquetes de Simyo ofrecen un poco más de flexibilidad que los de MásMóvil, si bien el precio final sería el mismo en ambos casos. Si se usara Simyo, el coste final de dotar de conectividad al prototipo no excedería los dos o tres euros. Con MásMóvil, el coste mínimo sería de tres euros, aunque con los servicios incluidos en esos tres euros sería más que suficiente.

3.4 Validación de resultados

Como se dijo en la Introducción, no sólo nos preocupa la eficiencia en el uso del agua que se logre en el pívot de riego gracias a nuestro prototipo, sino también la eficiencia misma de nuestro prototipo. Además, el rigor científico exige que todo resultado se someta a un cierto proceso de validación, de modo que aquello que se cree que es de una determinada manera o tiene una determinada propiedad, efectivamente sea así o tenga esa característica.

En nuestro caso particular, la eficiencia de nuestro prototipo está estrechamente relacionada con la cantidad de datos que se deben transmitir para poder controlarlo a distancia. A mayores datos transmitidos, mayor consumo (o necesidad de una tarifa más cara), y por lo tanto mayores costes de operación.

Para evaluar esto último, se han realizado una serie de pruebas de uso simuladas en las que se ha medido la cantidad de datos que se transfiere desde el prototipo hasta el terminal del usuario final (un PC). De este modo, podremos evaluar un poco mejor las tarifas anteriores y verificar si son válidas para dar conectividad a nuestro prototipo.

Se realizaron las siguientes mediciones:

- Transferencia de datos cuando el usuario final accede a la página del prototipo software: 115 KB.
- Si el usuario entra en la página del pívot y no se dispone de una cookie de autentificación no caducada, deberá hacer *login*, proceso que gasta 79 KB.
- Transferencia de datos mientras el usuario final mantiene abierta la página del prototipo software para monitorización del mismo. Se deben enviar todas las variables de estado de modo que en el terminal del usuario final se puedan actualizar los mapas y determinadas variables del panel de control: 527 bytes cada 5 segundos, es decir, 6 KB al minuto.
- Transferencia de datos cuando el prototipo envía un correo electrónico debido a que ha surgido algún evento de riesgo. Aproximadamente 3000 bytes. Esta medida es un poco más variable debido al protocolo STMP y dependiendo del tamaño del cuerpo del correo. Para dar un cierto margen, se considerará que su valor mínimo será de 4000 bytes.

Considerando las mediciones anteriores, el consumo de datos en algunas de las situaciones más comunes quedaría de la siguiente forma:

1) Supongamos que un usuario final típico, durante la época de riego, accede al prototipo web una media de 16 veces, es decir, cada hora y media. Esto supondría el consumo de 3.1 MB de datos por parte del prototipo según las cifras anteriores. Se ha considerado el caso más desfavorable, que es que cada vez que accede el usuario deba realizar el proceso de *login*.

2) Asumamos también que durante las 16 veces que accede, permanece conectado una media de 7 minutos para monitorizar su funcionamiento o modificar alguno de los parámetros de funcionamiento y, en este último caso, cerciorarse de que el cambio se ha producido. Esto nos dejaría un consumo de 672 KB.

3) Finalmente, consideremos que se produce al menos un evento de riesgo al día que provoca que un correo electrónico sea enviado para alertar al usuario final. Habría que añadir otros 4 KB al consumo. En ocasiones, el hecho de que se envíe un correo electrónico no significa que el evento de riesgo haya afectado al pívot, pues se pueden dar situaciones momentáneas que provoquen que se envíe el correo. Por ejemplo, supongamos que en una determinada ribera se ha autorizado el riego en un máximo de dos fincas al mismo tiempo. Es decir, pueden estar en funcionamiento dos pívots cualesquiera a la vez. Si en un determinado momento el agricultor que controla uno de los pívots en funcionamiento decide apagarlo porque considera que ya ha finalizado su riego, se producirá un aumento temporal de la presión de agua hasta que el agricultor de la siguiente finca encienda el suyo o el caudal de abastecimiento se auto-regule. En estos casos, aunque al usuario final le llegue un e-mail de alerta, no sería necesario un traslado inmediato a la finca. Primero, se optaría por una monitorización hasta verificar que efectivamente el evento de riesgo ha afectado al funcionamiento del pívot. Si es así, entonces ya sí que sería necesario desplazarse.

Todo lo anterior nos dejaría un total de 3.776 MB estimados por día de funcionamiento. Si se considera una campaña de riego de 20 días, tendríamos un total de 75.52 MB. Sumando un margen de un 30% de variabilidad en la estimación media global, el consumo medio final se quedaría en 98.176 MB.

Podemos concluir por lo tanto, que nuestro prototipo software ha sido diseñado correctamente, pues se le podría dar conectividad con cualquiera de las tarifas analizadas. De hecho, si se usara el de la compañía Simyo, los costes de operación debido a la conectividad no ascenderían a más de 1 euros por cada 20 días de riego.

4

Discusión y conclusiones

4.1 Discusión

En este trabajo se ha planteado el diseño y desarrollo de un prototipo que permitiera un manejo a distancia de los pívots de regadío en aras de un uso más eficiente del agua así como de evitar desplazamientos innecesarios para su reconfiguración y/o monitorización.

Como principales características de la solución a desarrollar se estableció que dispusiera de una interfaz web con un grado de usabilidad adecuado tanto en dispositivos fijos como móviles, que incorporase un sistema de alertas para eventos de riesgo y, finalmente, que la propia solución propuesta fuera eficiente en sí misma (costes de desarrollo, instalación y mantenimiento controlados).

Para lograr todo lo anterior, se hizo un estudio de las principales tecnologías existentes que pudieran ser empleadas en su desarrollo, capítulo 2. Se determinó que el prototipo hardware estaría basado en la plataforma Raspberry Pi complementada por la placa Custard Pi 2 y ciertos dispositivos menores, como un conector USB-Wifi, etc. En relación a este aspecto, se debe indicar que aunque la Raspberry Pi es aquella placa de hardware que más fácilmente cumple con los tres criterios fijados para la elección de las tecnologías de implementación (véase el plan de trabajo del apartado 1.3) – se dispone de capacidad técnica para su manejo pues su concepción original era meramente educativa, existe un conocimiento fácilmente accesible y suficientemente validado de la tecnología y su coste era asumible – la explosión del denominado hardware libre o hardware abierto (del inglés, *open hardware*) ha hecho que surjan multitud de alternativas. Cada una de estas alternativas busca una característica principal que la permite distinguirse del resto (tamaño, consumo, memoria, capacidad de procesamiento, interconexión con otros dispositivos, etc.) [62], [63].

La aparición de Arduino constituyó un punto y aparte en lo que a placas de hardware se refiere. Gracias a él, problemas de hardware bastante difíciles pasaron a ser problemas de software mucho más sencillos de resolver. Posteriormente surgió la Raspberry Pi y los horizontes se volvieron a ampliar. A partir de este momento, si uno buscaba desarrollar un proyecto de hardware simple se decantaba por Arduino, mientras que si necesitaba mayor capacidad de procesamiento o aplicaciones de alto nivel la elección caía del lado de la Raspberry Pi. Entre ellos y tras ellos han surgido otros proyectos – fuertemente

influenciados por ambos – de similares características, entre los que destacan los siguientes:

- **Tessel 2**: es una placa de hardware pensada para ser embebida en otros productos, facilitando así el Internet de las cosas (*Internet of things*). Sus principales características son que dispone de puerto Ethernet así como conector Wifi 802.11bng, una memoria RAM de 64 MB y una memoria Flash de 32 MB, un microprocesador Mediatek MT7620n a 580 MHz, y un co-procesador Atmel SAMD21 a 48 MHz para la gestión en tiempo real de las entradas y las salidas así como el consumo de potencia. Se alimenta mediante un conector microUSB. El sistema operativo de Tessel consiste en un intérprete de JavaScript. Por todas estas características, la Tessel 2 está a medio camino entre Arduino y las Raspberry Pi. No es un simple microcontrolador como Arduino, pues tiene un sistema operativo aunque muy ligero, pero tampoco pretende ser una competidora de las Raspberry Pi u otras placas computadoras completas basadas en Linux. Su precio es de unos 35 dólares estadounidenses [62], [64].
- **Wiring S:** el éxito de Arduino encumbró en gran medida esta excelente placa y su entorno de programación. Es muy similar a Arduino y su entorno de programación es compatible con cualquier hardware basado en la serie de procesadores AVR de Atmel (lo que incluye la propia línea de Arduino). Dispone de pines digitales de entrada y salida, pines analógicos de entrada, pines analógicos de salida tipo PWM (*Pulse Width Modulation*), pines de tipo TWI (*Two Wire Interface*) que facilitan la conexión de hasta 128 sensores/actuadores en una misma red – lo que a su vez permite que se comuniquen entre ellos, etc. Su microcontrolador es un poco más potente que el de Arduino.
- **Propeller ASC+:** la característica diferencial de esta placa es que posee ocho núcleos de procesado en su microcontrolador. Eso significa que se pueden ejecutar ocho procesos independientes de forma simultánea, respondiendo a sensores y otras entradas. Dependiendo de tu objetivo de desarrollo, si todos los procesadores se ponen a trabajar en paralelo en lugar de mediante interrupciones (cómo ocurriría si sólo estuviera disponible un procesador mononúcleo) el resultado puede ser tremendamente poderoso. Su precio ronda los 50 dólares estadounidenses. Se alimenta mediante un puerto USB de corriente o mediante una *jack* de alimentación

de 2.1 mm a entre 6 y 9 V. Su conexión con otros dispositivos es mediante una interfaz miniUSB [62], [66].

- **Netduino:** una placa microcontrolador muy similar a Arduino pero cuya diferencia con todas las anteriores es que emplea un microcontrolador STMicro STM32Fx de 32 bits (las anteriores empleaban 8 o 16 bits). Su sistema operativo es el NET Micro Framework y la tarjeta es directamente programable en C#. Se caracteriza por su potencia y flexibilidad. Constituye una alternativa similar a la Tessel 2, con la peculiaridad de emplear un microcontrolador de 32 bits en lugar de un microprocesador. Su precio es de 34.95 dólares estadounidenses [62], [67].
- **LaunchPad:** basada en el chip microcontrolador MSP430 de Texas Instruments es muy similar a Arduino con la principal diferencia de que su precio es únicamente de 10 dólares estadounidenses. Ofrece 14 pines de I/O y 15 KB de memoria programa, características muy básicas pero suficientes para muchos proyectos [62], [68]. Hasta no hace mucho, su principal desventaja es que no disponía de un IDE tan potente y cómodo como el de Arduino. Sin embargo, con la nueva plataforma de código abierto Energía esto ya ha quedado solucionado [69].
- **DisiSpark:** esta es otra tarjeta compatible con Arduino que se caracteriza por un tamaño muy reducido. Emplea el microcontrolador ATtiny85 y dispone de 6 pines I/O. Cuesta 9 dólares estadounidenses. Además, si se quiere aumentar sus prestaciones, está disponible una amplia variedad de kits de *shields* [64], [70].
- **Arduino Yun:** a partir del año 2012, más o menos, se empezó a observar una tendencia de las placas de computación y microcontrolador hacia lo inalámbrico. Ya no sólo se buscaba ser capaces de controlar algo, sino de hacerlo remotamente. Arduino Yun combina una tarjeta embebida basada en Linux y en el procesador AR9331, un Arduino clásico y un *shield* para Wifi. Por todo esto, su precio es de 69 dólares estadounidenses [63], [71].
- **Pinoccio:** financiado mediante un proyecto de *crowdfunding*, se constituye como una tarjeta compatible con Arduino que permite crear redes en malla basadas en el estándar IEEE 802.15.4. Incluye una batería de Li-Po y se le puede dotar de capacidad Wifi adicional por medio de un *shield*. La distancia máxima entre dos Pinoccio para que se puedan comunicar es de 100 metros. El paquete básico, que incluye una tarjeta Pinoccio y una batería LiPo, cuesta 59 dólares estadounidenses.

Si se quiere incluir en el paquete el *shield* para Wifi es precio asciende a 138 dólares estadounidenses [63], [72].

- **Geogram One:** es una placa compatible con Arduino cuyo objetivo son aplicaciones de rastreo y seguimiento. Integra un módem celular GSM y un receptor GPS. Ofrece además cuatro entradas analógicas y dos líneas digitales de I/O, así como un acelerómetro digital de 6 ejes. Su precio es de 120 dólares estadounidenses [63], [73].
- **BeagleBone:** seria alternativa a las Raspberry Pi por parte de Texas Instruments. A diferencia de esta, no fue pensada para fines educativos, sino que se diseñó desde cero para para comunicarse con otros componentes de hardware como sensores, actuadores, etc. El sistema operativo se almacena en una tarjeta SD, aunque dispone ranura para otra tarjeta microSD de 4GB. Se basa en el procesador AM335x ARM Cortes-A8 a 720 Mhz con 256 MB de memoria RAM y un acelerador para gráficos en 3D. Dispone de conector Ethernet y dos puertos USB. Aunque inicialmente su precio fue superior al de Raspberry, en la actualidad está en 45 dólares estadounidenses [63], [74].
- **pcDuino:** es otra tarjeta embebida que ejecuta Linux. Es compatible con numerosos de los *shields* de Arduino. Permite escribir código directamente en la tarjeta, como en un Arduino clásico, o desarrollar una aplicación de alto nivel sobre Linux. Está disponible a partir de 60 dólares estadounidenses. Se presenta como una opción atractiva por su facilidad y posibilidades de configuración. Incluye tanto CPU como GPU y las memorias RAM disponible van desde 256 MB hasta 1 GB. Todas las versiones llevan puerto Ethernet menos una que permite directamente conexión Wifi. La salida de video es HDMI y el almacenamiento mediante tarjetas SD [63], [75].
- **Gizmo:** todas las placas de computación descritas anteriormente se basaban en procesadores ARM. Sin embargo, ya han empezado a surgir alternativas basadas en procesadores X86. Gizmo es un ordenador de una sola placa extremadamente más flexible y poderoso que cualquiera de las tarjeas de microcontrolador anteriores. También dispone tanto de CPU como de GPU y salida HDMI. Su precio arranca en 199 dólares estadounidenses [63], [76].
- **Arduberry:** finalmente, mencionaremos la existencia de un *shield* muy particular que permite unir las dos principales referencias en lo que a tarjetas de

> microcontrolador – Arduino – y computación – Raspberry Pi – se refiere: Arduberry. Arduberry se presenta como un *shield* que ofrece una solución hardware simple y de bajo costo. El dispositivo se coloca sobre la Raspberry Pi y te permite añadir cualquier otro de los *shields* que emplea Arduino, incluido el propio Arduino. De esta forma, se tiene la capacidad de computación de una Raspberry pero con con la capacidad de interconexión con otros dispositivos electrónicos de un Arduino. Su precio es de 30 dólares estadounidenses. La parte negativa es que para poder sacarle partido debes adquirir también una Raspberry Pi y un Arduino (o *shields* de Arduino).

Hemos podido comprobar cómo existen multitud de alternativas en lo que a plataformas de hardware libre se refiere. Muchas de ellas hubieran cumplido sobradamente los requisitos que se impusieron en la Introducción para la elección de la tecnología del prototipo hardware. Destacar la versatilidad de la Tessel 2 o Netduino, que ofrecen lo mejor de una placa microcontrolador y lo mejor de una placa computadora pero con menos complejidad que las Raspberry Pi. Además, el precio de ambas es similar al de la Raspberry Pi. En caso de que en el futuro el proyecto necesitara mayor capacidad de procesamiento – por ejemplo, debido a que un mismo prototipo hardware controlase varios pívots de riego u otros elementos (bomba de agua, generador, etc.) además del pívot, la Propeller ASC+ sería la mejor elección por sus capacidades derivadas de su arquitectura multinúcleo. Si necesitáramos reducir costes, por ejemplo – para exportar el prototipo a países en vías de desarrollo o con pocos recursos – la LaunchPad de Texas Instruments sería la opción ideal. Además, su nuevo IDE Energía facilitaría enormemente el desarrollo de la parte software del prototipo y su convergencia con la parte hardware.

Como se indicó antes y durante los capítulos previos, en nuestro prototipo no sólo es importante la eficiencia de la parte hardware y software sino también su conectividad. Para ello, se decidió recurrir a las redes 3G/4G, aunque el Wifi sería lo mejor siempre que fuera factible. En este último caso, el emplear una placa como Pinoccio junto con Arduino nos permitiría crear redes *ad-hoc* que extendieran la señal desde un determinado punto de entrada. Esto podría ser realmente útil si estuviéramos ante una situación en la que, en una región relativamente próxima, existieran numerosos pívots de regadío controlados por distintas implementaciones de nuestro prototipo. Gracias a Pinoccio, sólo habría que suministrar red mediante Wifi (o quizá 3G/4G) a una de las placas Arduino. Después, serían las placas Pinoccio las que se encargarían de extenderla al resto de pívots y sus

respectivos prototipos de control, reduciendo así los costes de operación para todos estos sistemas de regadío controlados a distancia. La parte negativa de este enfoque reside en que, aunque se reducen los costes de operación debidos a la conectividad, aumentan los de implementación y desarrollo pues habría que adquirir tanto un Arduino como una placa Pinocco para cada prototipo. Además, la limitación de una distancia máxima de 100 metros entre cada placa parece que se quedaría insuficiente en muchos entornos geográficos. En cualquier caso, la idea de una red inalámbrica en malla o anillo es interesante y se podría explorar su implementación empleando la Raspberry Pi y los conectores USB-Wifi con enlaces punto a punto.

Si bien el sistema de alertas se estableció desde inicio en base a un sistema de envío de correos electrónicos – dado que este método se nutre igualmente de la red 3G o 4G que abastece el prototipo – otros sistemas de alerta concurrentes con este podrían ser de utilidad. Por ejemplo, mediante mensajes de texto (SMS, *Short Message Service*). Como se vio en el apartado 3.3, muchas OMV ofrecen este servicio de manera gratuita en muchas de sus tarifas. De este modo, con un sistema de alertas concurrente mediante correo electrónico y SMS nos aseguraríamos de que cualquier tipo de usuario final – independientemente de su grado de familiarización con las tecnologías web – quede alertado en caso de un evento de riesgo, pues aunque no dispusiera de correo electrónico es mucho menos probable que no dispusiera de un teléfono móvil con capacidad para recibir SMS. Esta solución no aumentaría (o lo haría muy levemente) los costes de operación pero si los de desarrollo. Para poder enviar los SMS desde el prototipo sería necesario no sólo un módem 3G/4G sino también un módem GSM/GPRS. Otra solución podría ser el uso de la placa Geogram One. Sin embargo, su elevado coste hace pensar que a priori sería más barato incluir en la Raspberry Pi un modem GSM/GPRS.

La placa BeagleBone sería equivalente a la Raspberry Pi y puede que incluso mejor desde un punto de visto tecnológico, pues se diseñó para aplicaciones reales desde cero y no con fines educativos. Además, el prototipo software diseñado para la Raspberry Pi sería igualmente válido en la BeagleBone. Todo esto facilitaría una futura transferencia tecnológica. Sin embargo, el emplear una placa que aunque cumpla nuestros criterios esté poco extendida en su uso (sobre todo si lo comparamos con la Raspberry Pi o Arduino) también tiene sus riesgos. Una comunidad de usuarios más reducida hace que las soluciones existentes que se pueden compartir entre los mismos se reduzcan también. Del mismo modo, tendremos a nuestra disposición menos complementos en caso de que no

podamos satisfacer nuestras necesidades con las prestaciones que la placa ofrece en su formato básico. Finalmente, si la placa de computación llegará a fracasar como producto comercial y nuestro prototipo estuviera basado en ella, a partir de ese momento nuestro prototipo estaría basado en una tecnología desactualizada y sin soporte, lo que lo haría inviable comercialmente. Todo esto es muy difícil que ocurriera con las Raspberry Pi.

Finalmente, y para terminar con la discusión acerca de las tecnologías empleadas en el prototipo hardware, indicar que en caso de que fuera necesaria una mayor complejidad en parte hardware del prototipo, pcDuino o Arduberry podrían ser el recurso. La ventaja de pcDuino frente a Arduberry es que su coste de adquisición sería menor. Sin embargo, el coste final de implementación ascendería en cualquiera de los dos casos ya que sería necesario adquirir *shields*, otras placas, etc.

En relación a las tecnologías del prototipo software, se decidió usar una combinación de PHP como principal lenguaje para desarrollar la aplicación complementado con Python para la lectura de los sensores. Si bien esta es una alternativa más que viable, como demuestra el apartado 3.4, bajo mi punto de vista lo ideal sería unificar todo el prototipo software bajo un mismo lenguaje. Es decir, usar un único lenguaje de programación con el que se pudiera dar salida a la aplicación web así como leer y controlar los dispositivos externos. De entre todos los lenguajes analizados en 2.3, Python sería el candidato ideal. De este modo, si el proyecto creciera en envergadura y en un futuro se necesitara una aplicación web más compleja y/o un control de más dispositivos electrónicos (sensores, actuadores, etc.), varios desarrolladores podrían trabajar independientemente garantizándose la convergencia de cada módulo de software, pues todos estarían implementados en Python (un desarrollador vería los módulos de los otros como cajas negras).

En tercer lugar, me gustaría discutir posibles mejoras del prototipo no necesariamente relacionadas con la parte tecnológica. En aras de fijar un objetivo abordable con un mínimo de garantías, se estableció que el prototipo fuera capaz de mejorar la eficiencia en el uso del agua para el riego y además que lo hiciera remotamente. Este primer objetivo parece totalmente coherente a la vista de la importancia que el agua tiene – tanto cualitativa como cuantitativamente – en la agricultura, recuérdese el capítulo de Introducción. Sin embargo, en la agricultura moderna influyen mucho otros muchos factores, tanto de tipo natural como artificial. Por ejemplo, el viento, el hielo, la temperatura, la humedad, si se utiliza o no algún tipo de abono o fertilizante, si se utiliza

o no algún tipo de pesticida o herbicida, el tipo de semilla empleada, condiciones en el momento de la siembra, etc. Controlar y modular de manera científica todas estas variables para maximizar el rendimiento de una finca o explotación agraria es algo que se escapa a mis conocimientos y que requeriría la incorporación de ingenieros agrarios al proyecto. No obstante, el prototipo construido puede servir de base para facilitar una gestión de las explotaciones agrarias basada en principios científicos y sobre todo mucho más eficiente (menos gasto de agua en el riego, menos consumo de combustible en la maquinaria, uso más controlado de herbicidas, pesticidas y abonos, etc.) a la par que cómoda para el agricultor.

Con este propósito, se podría complementar nuestro prototipo con una mini-estación meteorológica [77] que permitiera determinar el momento más apto para el riego en función de tres variables tan sencillas como son el viento, la temperatura y la humedad (o cantidad de lluvia). Aunque en los cultivos que se riegan mediante pívots no suele influir excesivamente, otros se ven afectados si reciben agua cuando la temperatura es muy alta o muy baja, por ejemplo. En base a estas tres variables se podría hacer que el prototipo envíe un email de aviso indicando que es el momento idóneo, conectara automáticamente el pívot (aunque esta opción seguramente no sería viable debido a la necesaria coordinación con otros agricultores durante la época de riegos) o que la propia aplicación web mostrara una estadística de estas tres variables. Con estas mejoras, el coste de implementación del prototipo aumentaría, pero también lo harían sus prestaciones. En caso de que no fuera viable adherir la mini-estación meteorológica, se podría plantear que el prototipo tuviera una conexión permanente con un servidor que almacenara los datos meteorológicos actualizados de una determinada zona. Desde hace años, la Agencia Estatal de Meteorología (AEMET) dio acceso libre y gratuito a todos sus datos por medios electrónicos (tanto en formato web bajo el protocolo HTTP como en formato servidor de ficheros bajo el protocolo FTP) [78], [79]. Esta solución aumentaría los costes de operación derivados de la conectividad, pues el consumo de datos sería mayor, y tiene como desventaja respecto a la mini-estación meteorológica que la medidas meteorológicas serían mucho menos precisas (la AEMET dispone de varias estaciones meteorológicas por provincia, pero su localización puede distar bastante de la del pívot). En cualquier caso, seguiría siendo un buen complemento de nuestro prototipo base.

El disponer de esta tecnología sería de tremenda utilidad para el estudio o investigación de muchos aspectos de la ingeniería agraria: contraste de semillas, contraste

de abonos o simplemente analizar como varía el rendimiento de una explotación agraria a otra en función del *modus operandi* de cada agricultor. Asimismo, se podría plantear la extrapolación de nuestro prototipo a otros campos de la ingeniería agropecuaria. Una primera extensión muy sencilla sería para el control de otros sistemas de regadío similares, como el cañón de riego. También para otros más complejos como riego por goteo o por aspersión, típicos de invernaderos, explotaciones hortofrutícolas o plantaciones de viña. De la misma manera, se podría usar en explotaciones ganaderas para tareas como llenado automático de los bebederos de los animales (cuando se detectara que el nivel está por debajo de un determinado umbral), etc. Con todo, la utilidad de este tipo de tecnología en las explotaciones ganaderas es más limitada, pues el cuidado de los animales precisa de muchas tareas que es casi imposible automatizar o controlar a distancia.

Para terminar esta discusión, comentar ciertas limitaciones de este TFM que merecen una consideración. Como ocurre en muchos otros proyectos, el mayor factor limitante para el desarrollo de este TFM ha sido el tiempo disponible (y la capacidad de desarrollo en este tiempo). Hubiera sido tremendamente interesante haber desarrollado distintas implementaciones de nuestro prototipo – por ejemplo, un basada en la Tessel 2 con el actual prototipo de software (PHP+Python+jQueryMobile), otra basada en Netduino con la implementación de la aplicación web en C# más los sensores controlados por Python, otra basada en la Raspberry Pi con todo el prototipo software escrito en Python, o una versión de bajo coste que empleara la placa LaunchPad – y haberlas comparado, al menos en condiciones de laboratorio y/o simulación. Así, se podría contrastar el diferente grado de idoneidad de cada una para una futura transferencia tecnológica de nuestro prototipo al área comercial.

De igual forma, hubiera sido muy útil una validación del prototipo en condiciones reales, pero sobre todo, una validación prolongada en condiciones reales que nos hubiera servido como fuente de retroalimentación desde un usuario final real. Con ella podríamos responder a preguntas como: ¿hemos considerado todos los eventos de riesgo posibles (desconexión imprevista, presión de abastecimiento inapropiada)?, ¿el sistema de alertas mediante e-mails es de utilidad, se puede suprimir, o que hay configurar otro nuevo sistema de alertas?, ¿las estimaciones de uso medio de datos transmitidos por día en campaña de riego son aceptables?, ¿la cobertura disponible de las operadoras estudiadas permite un control remoto del prototipo o existen puntos de sombra sin conectividad a los

que habrá que dar un solución *ad-hoc*?, etc. Se propuso un primer intento de validación en condiciones reales durante la campaña de regadío de la primavera de este año (cosecha de cereales de verano). Sin embargo, por razones externas a este TFM no fue posible. Se volverá a intentar durante la campaña de regadío de verano (cultivos de girasol y maíz), pero tampoco está garantizado.

4.2 Conclusiones

En este TFM se ha demostrado que se puede ofrecer una solución innovadora a la tecnología de regadío mediante pívots que permita un uso más eficiente del agua en la agricultura al mismo tiempo que evita desplazamientos innecesarios derivados de la monitorización o puesta en marcha de esta nueva tecnología agraria.

Su diseño se dividió en dos partes fundamentales: la parte hardware y la parte software. La primera se basó en la placa computadora Raspberry Pi junto con la placa Custad Pi 2, dado que de esta forma se facilitaba tanto el control de dispositivos externos (sensores, actuadores y autómatas del pívot de regadío) como el desarrollo de una aplicación de usuario a alto nivel. En la parte software se empleó el lenguaje de programación PHP para la aplicación de usuario y el lenguaje de programación Python para la lectura de los dispositivos externos. En su diseño se tuvieron en cuentas aspectos como interoperabilidad para dispositivos fijos y móviles – que se logró mediante jQuery Mobile – un grado de usabilidad adecuado a las capacidades técnicas esperadas del usuario final de nuestro prototipo y la mayor eficiencia posible en la cantidad de datos transmitidos entre el prototipo y el dispositivo de control del usuario final (PC, Smartphone, tableta, etc.) durante el control remoto del mismo. Tras implementar este diseño y realizar una primera toma de medidas en condiciones simuladas se verificó que nuestro prototipo cumplía con las propiedades esperadas de él, pues la transferencia de datos durante el uso medio estimado en un día de funcionamiento no supera el megabyte.

Derivado de la localización de los entornos en el que se espera que nuestro prototipo entre en funcionamiento, para su conectividad se optó por usar la red 3G/4G. Tras realizar un estudio del mapa de cobertura en la Comunidad Autónoma de Castilla y León, se determinó que las dos mejores coberturas son las que ofrecen Vodafone y Orange, si bien no existe mucha diferencia entre compañías. Finalmente, el análisis de la oferta de

servicios de las principales OMV posicionó a la Simyo y MásMóvil como las mejores candidatas con las que contratar el servicio de conexión, facilitando así el paso al mundo comercial de nuestro prototipo.

En resumen, la inclusión de las TIC en un sector como el agrario puede constituir un factor diferencial en el modo y costes de explotación de las tierras de regadío así como contribuir a reducir la huella medio ambiental derivada de la actividad humana.

Referencias:

[1] Resumen del 2º informe de las Naciones Unidas sobre el desarrollo de los recursos hídricos en el mundo, realizado por GreenFacts, 2009.

[2] Informe de las Naciones Unidas sobre el desarrollo de los recursos hídricos en el mundo – Resumen para el World Water Assessment Programme, '*Agua para todos Agua para la vida*,' título original: '*Water for people, water for life*', primera edición impresa en 2003, UNESCO, Paris, Francia.

[3] Simonovic S.P., 'Worls water dynamics: global modeling of water resources,' *Journal of Environmental Management*, vol. 66, nº 3, pp.: 249-267, Nov. 2002.

[4] H. Turral, J. Burke, J-M. Faurès, FAO Water Report 36, '*Climate change, water and food security*', 2011, Roma.

[5] ONU, Nota informativa: 'Agua y agricultura en la economía verde', dentro del Programa ONU-Agua para la Promoción y la Comunicación en el marco del Decenio (UNW-DPAC).

[6] FAO, '*Descubrir el potencial del agua para la agricultura*,' 2003.

[7] Web de la Organización de las Naciones Unidas, '*Agua: ¿Por qué actúa la ONU?*', disponible en: http://www.un.org/es/globalissues/water/, Abril 2015.

[8] Web de la Organización de las Naciones Unidas para la Alimentación y la Agricultura, '*La labor de la FAO en el agua*', disponible en: http://www.fao.org/water/es/, Abril 2015.

[9] Web de la Organización de las Naciones Unidas para la Educación la Ciencia y la Cultura, '*El agua dulce*', disponible en: http://www.unesco.org/new/es/natural-sciences/environment/water/, Abril 2015.

[10] Web de Water for People, disponible en: http://www.waterforpeople.org/about/, Mayo 2015.

[11] Mensaje del Secretario General de la ONU el día mundial del agua 2015, disponible en: http://www.un.org/es/events/waterday/2015/sgmessage.shtml, Mayo 2015.

[12] Mazoyer M y Roudart L., 'A history of world agriculture: from the Neolithic age to the current crisis,' *Journal of Historical Geography*, vol. 34, nº 2, pp.: 375-376, Apr. 2008.

[13] Mencher J.P., '*Antropology and human deveplopment*,' disponible en: http://www.eolss.net/EolssSampleChapters/C11/E6-60-01-01/E6-60-01-01-TXT-02.aspx#ANTHROPOLOGY_AND_HUMAN_DEVELOPMENT_, Junio 2015.

[14] '*Agriculture as "Worst Mistake in the History of the Human Race"?*', disponible en: http://www.livinganthropologically.com/anthropology/agriculture-as-worst-mistake-in-the-history-of-the-human-race/, Junio 2015.

[15] FAO, 'Una oportunidad para aprovechar: agua en la agricultura,'.

[16] Programa de ONU-Agua para la Promoción y la Comunicación en el marco del Decenio (UNW-DPAC), '*Agua y agricultura en la economía verde*'.

[17] Programa de Naciones Unidas para el Medio Ambiente (UNEP), '*Water and the Green Economy: capacity development aspects*,' 2012.

[18] FAO, 38 Informe sobre temas hídricos, '*Afrontar la escasez de agua: un marco de acción para la agricultura y la seguridad alimentaria*,' Roma, 2013.

[19] Nota informativa de la Conferencia de las Naciones Unidas sobre el Desarrollo Sostenible RIO+20, dentro del Programa de ONU-Agua para Promoción y la Comunicación en el marco del Decenio (UNW-DPAC), '*Agua y tecnología en la transición hacia una economía verde*'.

[20] Raspberry Pi Foundation, '*What is Raspberry Pi?,*' disponible en: https://www.raspberrypi.org/help/what-is-a-raspberry-pi/, Julio 2015.

[21] Arduino Genuino, '*What is Arduino?*,' disponible en: https://www.arduino.cc, Julio 2015.

[22] Bourque B., '*Arduino VS Raspberry Pi: mortal enemies, or best friend?*,' disponible en: http://www.digitaltrends.com/computing/arduino-vs-raspberry-pi/, Julio 2015.

[23] Hacedores Maker Community, '*Arduino o Raspberry Pi, ¿cuál es la mejor herramienta para tí?*,' disponible en: http://hacedores.com/arduino-o-raspberry-pi-cual-es-la-mejor-herramienta-para-ti/, Julio 2015.

[24] Raspberry Pi Foundation, sección de descargas, disponible en: https://www.raspberrypi.org/downloads/, Julio 2015.

[25] Web oficial de Raspbian, disponible en: http://www.raspbian.org, Julio 2015.

[26] Suehle R., '*Should I get an Arduino or a Raspberry Pi?,*' disponible en: http://opensource.com/life/15/5/should-i-get-arduino-or-raspberry-pi, Julio 2015.

[27] Doutel F., '*Probamos la nueva Raspberry Pi 2: a fondo,*', disponible en: http://www.xatakahome.com/trucos-y-bricolaje-smart/probamos-la-nueva-raspberry-pi-2-a-fondo, Julio 2015.

[28] Raspberry Pi Foundation, 'GPIO: Raspberry Pi models A and B,' disponible en: https://www.raspberrypi.org/documentation/usage/gpio/, Julio 2015.

[29] SF Innovations, 'Having fun with electronics and the Raspberry Pi,' disponible en: http://www.sf-innovations.co.uk/custard-pi-1.html, Julio de 2015.

[30] SF Innovations, 'Full Technical Documentation of Custard Pi 2,' disponible en: http://www.sf-innovations.co.uk/downloads/cpi1doc.pdf, Agosto 2015.

[31] Web de la comunidad PHP, 'Introducción,' diposnible en: http://php.fubra.com/manual/es/intro-whatis.php, Julio 2015.

[32] Web de la comunidad PHP, 'Qué puede hacer PHP,' disponible en: http://php.fubra.com/manual/es/intro-whatcando.php, Julio 2015.

[33] Oracle, 'JavaServer Pages Technology,' disponible en: http://www.oracle.com/technetwork/java/javaee/jsp/index.html, Julio 2015.

[34] IBM Knowledge Center, 'Arquitectura JPA (Java Persistence API),' disponible en: http://www01.ibm.com/support/knowledgecenter/SSEQTP_7.0.0/com.ibm.websphere.nd.doc/info/ae/ae/cejb_persistence.html, Julio 2015.

[35] Python Organization, 'Beginners Guide/Overview,' disponible en: https://wiki.python.org/moin/BeginnersGuide/Overview, Julio 2015.

[36] Ruby, el mejor amigo de un desarrollador, 'Acerca de Ruby,' disponible en: https://www.ruby-lang.org/es/about/, Julio 2015.

[37] WAMPServer, Apache, PHP, MySQL sous Windows, disponible en: http://www.wampserver.com, Julio 2015.

[38] Apache Tomcat, disponible en: http://tomcat.apache.org, Julio 2015

[39] Django Project, disponible en: https://www.djangoproject.com, Julio 2015

[40] Ruby on Rails, disponible en: http://rubyonrails.org, Julio 2015.

[41] Cruz, J. 'Java ME 8 + Raspberry Pi + Sensors = IoT World (Part 1),' disponible en: http://www.oracle.com/technetwork/articles/java/cruz-gpio-2295970.html, Agosto 2015.

[42] Cruz, J. 'Java ME 8 + Raspberry Pi + Sensors = IoT World (Part 2),' disponible en: https://community.oracle.com/docs/DOC-914698, Agosto de 2015

[43] Estándar internacional para la evaluación de calidad de software ISO/IEC 9126.

[44] Estándar internacional ISO/IEC 9241.

[45] Web de jQueryMobile, disponible en: http://jquerymobile.com/about/, Julio 2015.

[46] Operadoras Móviles Virtuales (OMV), disponible en: http://operadoras-moviles.com/omvs/, Agosto 2015.

[47] Pau B., 'Todas las compañías de móviles de España, ¿cuál es la mejor?,', disponible en: http://www.comparaiso.es/moviles/guias/companias-de-moviles-lista-completa, Agosto 2015.

[48] Mapas de cobertura de Orange, disponible en: http://4g.orange.es/cobertura/, Agosto 2015.

[49] Mapas de cobertura de Vodafone, disponible en: http://www.vodafone.es/conocenos/es/vodafone-espana/mapa-de-cobertura/consulta-de-cobertura-movil/, Agosto 2015.

[50] Mapas de cobertura de Movistar, disponible en: http://www.movistar.es/particulares/coberturas/movil/, Agosto 2015.

[51] Mapas de cobertura de Yoigo, disponible en: http://cobertura.yoigo.com, Agosto 2015.

[52] *OpenSignal, find the best network in your area*, disponible en: http://opensignal.com/coverage-maps/Spain/, Agosto 2015. (Mapa centrado en la Comunidad Autónoma de Castilla y León)

[53] Comparador de tarifas "Comparaiso", disponible en: http://www.comparaiso.es/moviles/tarifas/solo-sim/, Agosto 2015.

[54] Comparador de tarifas "Kelisto", disponible en: http://telefonia-movil.land.kelisto.es/, Agosto de 2015.

[55] Comparador de tarifas "Rastreator", disponible en: https://tarifasmovil.rastreator.com/datos-comparativa.aspx#Q1, Agosto 2015.

[56] Tarifas prepago de Simyo, disponible en: http://www.simyo.es/tarifas/prepago.html, Agosto 2015.

[57] Tarifa de Lowi, disponible en: https://www.lowi.es/tarifas-movil/, Agosto 2015.

[58] Tarifas para navegar pre-pago de Lowi, disponible en: https://masmovil.es/es/tarifas-navegar-prepago/, Agosto 2015.

[59] Tarifas de PepePhone, disponible en: https://www.pepephone.com/movil/tarifas, Agosto 2015.

[60] Tarifas de Movizelia, disponible en: https://www.movizelia.com, Agosto 2015.

[61] Tarifas de Amena, disponible en: http://www.amena.com/tarifas/, Agosto 2015.

[62] Hacedores Maker Community, '¿Qué tarjeta de desarrollo elegir? (Parte 1),' disponible en: http://hacedores.com/que-tarjeta-de-desarrollo-elegir-parte-1/, Junio 2014. Último acceso: Agosto 2015.

[63] Hacedores Maker Community, '¿Qué tarjeta de desarrollo elegir? (Parte 2),' disponible en: http://hacedores.com/que-tarjeta-de-desarrollo-elegir-parte-1/, Junio 2014.

[64] Página oficial del proyecto Tessel 2, disponible en: https://tessel.io, Agosto 2015.

[65] Página oficial del proyecto Wiring, disponible en: http://wiring.org.co/hardware/, Agosto 2015.

[66] Web oficial del fabricante de la Propeller ASC+, disponible en: https://www.parallax.com/product/32214, Agosto 2015.

[67] Web oficial del proyecto Netduino, disponible en: http://www.netduino.com, Agosto 2015.

[68] Web oficial de TI LaundhPad de Texas Instruments, disponible en: http://www.ti.com/ww/en/launchpad/about.html, Agosto 2015.

[69] Web oficial del proyecto energía, disponible en: http://energia.nu, Agosto 2015.

[70] Web oficial de Digistump, disponible en: http://digistump.com/category/1, Agosto 2015.

[71] Web oficial del proyecto Arduino, Arduino Yún, disponible en: https://www.arduino.cc/en/Main/ArduinoBoardYun?from=Main.ArduinoYUN, Agosto 2015.

[72] Web oficial del proyecto Pinoccio, disponible en: https://pinocc.io, Agosto 2015.

[73] Web oficial del proyecto Geogram One, disponible en: http://dsscircuits.com/sale/product/dssc0120, Agosto 2015.

[74] Web oficial del proyecto BeagleBone, disponible en: http://beagleboard.org/bone, Agosto 2015.

[75] Web oficial del proyecto pcDuino, disponible en: http://www.pcduino.com, Agosto 2015.

[76] Web oficial del proyecto Gizmo, disponible en: http://www.gizmosphere.org/products/gizmo-2/, Agosto 2015.

[77] Mini-estación meteorológica comercial para ayuda a los sistemas de riego frente al viento, la lluvia y las bajas temperaturas. Disponible en: http://www.hunterindustries.com/es/product/sensores/mini-estacion-meteorologica, Agosto 2015.

[78] Datos meteorológicos disponibles en la web de la Agencia Estatal de Meteorología, Ministerio de Agricultura, Alimentación y Medio Ambiente del Gobierno de España, disponible en: http://www.aemet.es/es/eltiempo/observacion/ultimosdatos y en ftp://ftpdatos.aemet.es/, Agosto 2015.

[79] Noticia sobre el acceso libre y gratuito a los datos de la Agencia Estatal de Meteorología, disponible en: http://www.tiempo.com/ram/12029/aemet-establece-el-acceso-libre-y-gratuito-a-todos-sus-datos-por-medios-electronicos/, Agosto 2015.