CEA Comité Español de Automática
XLVI JORNADAS AUTOMÁTICA CARTAGENA 2025
UNIVERSIDADE DA CORUÑA

Jornadas de Automática

SCADA basado en software abierto para planta demostrativa de desalación

Poyatos-Bakker, Aarón-Raúl[a,*], Martínez-Roa, Antonio[b], Roca, Lidia[a], Palenzuela, Patricia[a], Gil, Juan D.[b]

[a] *CIEMAT-Plataforma Solar de Almería-CIESOL, Ctra. de Senés s/n, Tabernas, 04200, Almería, España.*
[b] *Departamento de Informática, CIESOL-ceiA3, Universidad de Almería, Ctra. Sacramento s/n, 04120, Almería, España.*

To cite this article: Poyatos-Bakker, Aarón-Raúl, Martínez-Roa, Antonio, Roca, Lidia, Palenzuela, Patricia, Gil, Juan D. 2025. Open-source web-based SCADA system for a demonstrative desalination plant. Jornadas de Automática, 46. https://doi.org/10.17979/ja-cea.2025.46.12195

Resumen

La destilación por membranas se presenta como una tecnología con gran potencial para concentrar la salmuera procedente de ósmosis inversa, convirtiendo así a la tecnología de desalación más utilizada en la actualidad en un proceso más sostenible. Con el fin de optimizar el proceso de concentración de salmueras, es necesario disponer de estrategias de control y un sistema de supervisión adecuado que permita mantener las condiciones de operación estables. Este trabajo presenta el desarrollo de un sistema de supervisión, control y adquisición de datos (SCADA) de una planta de destilación por membranas a escala demostrativa, ubicado en las instalaciones de la infraestructura Agroconnect de la Universidad de Almería. En concreto, se propone un sistema SCADA del control regulatorio del caudal de la planta, el cual se encuentra integrado en una red de contenedores Docker basado en software abierto, ofreciendo esto una serie de ventajas en términos de virtualización.

Palabras clave: Supervisión de procesos, Control de procesos, Adquisición de datos de sensores remotos, Control y operación óptimos de sistemas de recursos hídricos, Sistemas de instrumentación y control

Open-source web-based SCADA system for a demonstrative desalination plant

Abstract

Membrane distillation is emerging as a promising technology for concentrating brine from reverse osmosis, thus making the most widely used desalination method today more sustainable. To optimize the brine concentration process, it is essential to implement control strategies and an appropriate monitoring system to maintain stable operating conditions. This work presents the development of a supervisory control and data acquisition (SCADA) system for a membrane distillation plant at a demonstrative scale, located at the Agroconnect infrastructure of the University of Almería. Specifically, a SCADA system is proposed for regulatory control of the plant's flow rate, which is integrated into a Docker container network based on open-source software, offering several advantages in terms of virtualization.

Keywords: Process supervision, Process control, Remote sensor data acquisition, Optimal control and operation of water resources systems, Instrumentation and control systems

1. Introducción

La tecnología de ósmosis inversa (RO, por sus siglas en inglés *Reverse Osmosis*) está considerada como la tecnología referente para la desalación de agua de mar (Qasim et al., 2019). No obstante, para hacer esta tecnología más sostenible sería conveniente tratar la salmuera resultante (Andrés-Mañas et al., 2025), ya que su vertido al mar produce un impacto negativo sobre el medioambiente (Ahmad and Baddour, 2014).

La destilación por membranas (MD, por sus siglas en inglés *Membrane Distillation*) es un método de desalación térmica que se presenta como un buen candidato para concentrar dicha salmuera hasta niveles de concentración elevados, de forma que se reduzca su volumen considerablemente (Mar-

*Autor para correspondencia: aaron.pb@psa.es

tinetti et al., 2009; Viader et al., 2021) y que, posteriormente, se pueda valorizar mediante un proceso de cristalización. El proceso de MD es un proceso de separación térmica que tiene lugar a través de una membrana microporosa e hidrofóbica que solo deja pasar el vapor. El vapor de agua se consigue por una diferencia de presión al acercar dos flujos de agua a distintas temperaturas (Alkhudhiri et al., 2012), produciéndose el permeado cuando el vapor entra en contacto con el canal de condensación. Existen varias configuraciones de MD dependiendo del modo de extracción del permeado y el diseño de los módulos que contienen la membrana hidrofóbica (Shalaby et al., 2022).

Por otra parte, este tipo de tecnologías requieren unas condiciones de operación estables de caudal y temperatura para funcionar de manera óptima. Es en este contexto donde las estrategias de control son necesarias para automatizar y optimizar los parámetros críticos del proceso (Gil et al., 2018). Además, la gestión y monitorización de datos en tiempo real es otro factor esencial para analizar el funcionamiento del sistema. Como se señala en Phuyal et al. (2020), la implementación de un sistema de supervisión, control y adquisición de datos (SCADA, por sus siglas en inglés *Supervisory Control And Data Acquisition*) permite gestionar y controlar cualquier sistema local o remoto mediante una interfaz gráfica que comunica al usuario con el sistema.

La necesidad de un SCADA implica el desarrollo tanto de la interfaz disponible e interactuable para el usuario (conocido como *frontend*) como la gestión interna de variables, su registro en bases de datos y mapeado a los elementos presentados en la interfaz (conocido como *backend*). La elección de la herramienta SCADA a utilizar va ligado habitualmente a las especificaciones del *hardware* y el sistema operativo en el que se va a alojar el *backend*. Esto supone un punto de inflexión, ya que la mayoría de sistemas SCADA son software propietario, desarrollados comúnmente para sistemas operativos de Microsoft, suelen ser altamente costosos y complicados de mantener (Babayigit and Abubaker, 2023).

En la actualidad, se estudian alternativas que velan por el desarrollo de software abierto (conocido como *open-source*) basados en la web, que tratan de reducir esa dependencia con ciertos sistemas operativos o del *hardware* utilizado en la arquitectura del sistema y ofrecen compatibilidades con varios protocolos de comunicación habituales en la industria (como OPC y MQTT) (Lahti et al., 2011). Una alternativa bastante prometedora es el uso de Docker para virtualizar sistemas distribuidos (Sollfrank et al., 2020). Concretamente, el uso de la herramienta `docker-compose` permite el despliegue de una red de contenedores virtuales que operan de forma aislada, creando un entorno capaz de gestionar múltiples componentes (Ibrahim et al., 2021), como es el caso en sistemas SCADA. Con `docker-compose` se tiene total independencia con las características del *host* donde se aloja la red, debido a las características de aislamiento mencionadas. Es fácil de mantener, actualizar y escalar al presentar una estructura modular por contenedores, pudiendo gestionar estos sin necesidad de interrumpir toda la red y evaluar el rendimiento tanto general como por contenedor.

En el presente trabajo se tiene por objeto el desarrollo de un sistema SCADA de una planta MD a escala demostrativa, utilizando un software *open-source* basado en la web. En dicho SCADA, se desea integrar todas las estrategias de control regulatorio que permitan mantener el proceso estable. Tanto los controladores como el SCADA se integran en una red de contenedores Docker, lo que garantiza su funcionamiento de manera aislada respecto al hardware y sistema operativo en el que se aloje.

2. Materiales y métodos

2.1. Descripción de la planta MD

La planta piloto MD utilizada en este trabajo se muestra en la Figura 1 y se ubica en el Instituto Andaluz de Investigación y Formación Agraria, Pesquera, Alimentaria y de la Producción Ecológica (IFAPA) cerca de la Universidad de Almería (UAL). El trabajo se desarrolla en el marco del proyecto Agroconnect (`www.agroconnect.es`), un proyecto colaborativo entre IFAPA y la UAL.

Figura 1: Planta piloto de destilación por membranas en IFAPA.

Entre las instalaciones del proyecto, se dispone de una red integral de agua que utiliza una planta de RO como principal unidad de desalación (Gil et al., 2024) y la planta MD para procesar el rechazo de salmuera producido por la RO (Andrés-Mañas et al., 2025).

La planta de la Figura 1 se compone de 10 módulos configuración de hueco de aire con vacío (VAGMD, de sus siglas en inglés *Vacuum Air Gap Membrane Distillation*). También dispone de dos intercambiadores de calor para que los fluidos de alimentación al módulo intercambien el calor con los fluidos caliente y frío. En la Figura 2 se muestra el diagrama de funcionamiento de la planta MD. En Andrés-Mañas et al. (2025) se detalla el modo de operación *batch* de la planta MD y el suministro de frío y calor al sistema. El aporte de energía térmica se realiza a través de un campo solar de captadores planos con reflectores.

Respecto a la comunicación, se tiene acceso al PLC integrado de la planta mediante un servidor OPC-UA (de sus siglas en inglés *Open Platform Communications Unified Architecture*) configurado en un PLC secundario. Este se comunica a su vez vía Modbus con el PLC integrado. Esta arquitectura permite la comunicación remota con la planta, facilitando la incorporación de mecanismos adicionales en el control, como el mecanismo *anti-windup*, que no son posibles de configurar de manera local en el PLC integrado (Sánchez et al., 2024).

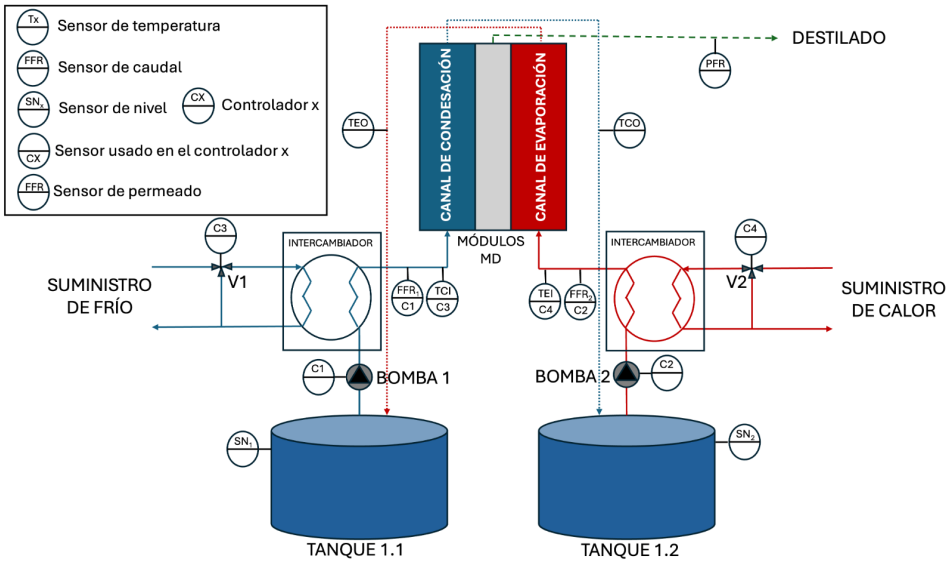


Figura 2: Diagrama esquemático de la instalación MD en IFAPA (Sánchez et al., 2024).

2.2. *Software FUXA*

FUXA es un software de visualización de procesos basado en la web. Ofrece varias posibilidades de uso, como SCADA, HMI (por sus siglas en inglés, *Human Machine Interface*) o un panel de visualización de datos, también conocido en inglés como *dashboard*, así como una amplia gama de protocolos de comunicación comunes en la industria: Modbus RTU/TCP, Siemens S7 Protocol, OPC-UA, BACnet IP, MQTT y Ethernet/IP. Es un software *open source* disponible públicamente en el repositorio GitHub: `https://github.com/frangoteam/FUXA`.

FUXA dispone de un editor visual e interactivo para diseñar y configurar tanto la parte *frontend* como *backend*, incluyendo elementos visuales predefinidos, la posibilidad de desarrollar scripts en javascript para procesos más particulares, gestión de alarmas y perfiles de acceso de usuarios para proporcionar una capa de seguridad.

2.3. *Red de contenedores Docker con docker-compose*

Los controladores de la planta MD se desarrollan en un contenedor Docker, para lo cual se han empleado las herramientas de construcción de imágenes vía `Dockerfile`.

Para crear la red de contenedores, se ha utilizado la herramienta `docker-compose`. Dicha herramienta permite la ejecución y gestión de múltiples contenedores. El montaje de la red se realiza de manera sencilla desde un archivo en formato `YAML`, en el que se pueden configurar los contenedores a generar y ejecutar, incluyendo propiedades en el arranque como volúmenes, puertos o redes internas asociadas. Incluso, es posible que la red gestione la propia construcción de la imagen y la generación posterior del contenedor, al indicar directamente la ruta local al proyecto en el que se encuentra el archivo `Dockerfile`. Los contenedores se pueden configurar para que siempre se vuelvan a iniciar en caso de quedarse bloqueados o que el sistema los interrumpa.

A continuación se señalan algunos de los comandos habituales de `docker-compose` para resaltar la flexibilidad y sencillez que se tiene para gestionar la red.

```
# Levantar la red
docker compose up
# Levantar la red reconstruyendo las imagenes
docker compose up --build
# Levantar la red pero solo con algunos servicios
docker compose up fuxa
# Tumbar la red
docker compose down
```

Código 1: Comandos de ejemplo de `docker-compose`.

2.4. *Controlador PID con mecanismo anti-windup*

Para el sintonizado de controladores, se ha considerado la función de transferencia $C(s)$ ideal de un controlador PI:

$$C(s) = \frac{U(s)}{E(s)} = K_p\left(1 + \frac{1}{T_i\, s}\right), \tag{1}$$

siendo K_p la ganancia proporcional y T_i el tiempo integral. Además, $U(s)$ representa la entrada del sistema y $E(s)$ la señal de error.

Los parámetros del controlador $C(s)$ se han obtenido empleando el método SIMC (Skogestad and Grimholt, 2012), que es recomendable tanto para estrategias de seguimiento como de rechazo frente a perturbaciones con tiempos de retardo considerables. Las siguientes ecuaciones determinan los valores de los parámetros K_p y T_i para un sistema que sigue un modelo dinámico $G(s)$ lineal de primer orden con retardo, presentada en la Ecuación (3):

$$T_i = \text{mín}\left(\tau + \frac{\theta}{3},\ 4(\tau_c + \theta)\right), \tag{2a}$$

$$K_p = \frac{1}{k}\frac{3\tau + \theta}{3(\tau_c + \theta)}, \tag{2b}$$

con ganancia estática k, constante de tiempo τ, tiempo de retardo θ y constante de tiempo en bucle cerrado τ_c:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{k}{\tau s + 1}\, e^{-\theta s}. \tag{3}$$

Se han incluido además límites de saturación, un mecanismo *anti-windup* con ganancia $1/\sqrt{T_i}$ y unas bandas de histéresis para evitar cambios frecuentes en los actuadores.

2.5. *Procedimiento de los ensayos en lazo abierto y lazo cerrado*

Para caracterizar la dinámica de cada bomba como un modelo lineal de primer orden con retardo, véase la Ecuación (3), se ha realizado un ensayo en bucle abierto para analizar la respuesta del sistema. En el ensayo, se aplican una serie de saltos del 5 % en los puntos de operación de las bombas, utilizando intervalos de tiempo suficientes para que el sistema alcance el estado estacionario. Como la planta MD va a operar con caudales entre 3 y 5 m^3/h, el ensayo se limita en el rango de operación del 20 % al 40 %, con cambios en escalón tanto de subida como de bajada.

Después, para cada cambio, se determinan los parámetros de $G(s)$. Para estimar el tiempo de retardo, θ, se ha tomado como criterio el intervalo de tiempo en el que se logra alcanzar el 5 % de la señal de salida del sistema, $Y(s)$. Una vez obtenidas todas las funciones de transferencia en los diferentes puntos de operación, se calcula una función de transferencia de parámetros promedio, la cual se usará como función nominal para el cálculo del controlador.

Posteriormente, se han efectuado dos tipos de ensayo en bucle cerrado. Uno preliminar desde MATLAB y otro ensayo utilizando el SCADA desarrollado. En ambos, se han seguido los pasos mostrados en la Tabla 1. Se ha tenido en cuenta las condiciones de arranque del sistema (Andrés-Mañas et al., 2025) y la transferencia sin salto al control automático.

Tabla 1: Pasos efectuados considerando el modo de arranque de la planta MD en manual y el salto de transferencia a los controladores. Se han introducido los mismos *setpoints* en ambas bombas.

Nº	**Fase**	***Setpoint***	**Tiempo** [min]
1	Arranque manual de bombas	20 [%]	1
2	Estabilización de caudales	30 [%]	1
3		40 [%]	1
4	Bajada a punto de partida	22 [%]	1
5	Transferencia a automático	3 [m^3/h]	5
6	Operación deseada de la planta	5 [m^3/h]	5
7	Otros cambios de referencia	4 [m^3/h]	5
8		3 [m^3/h]	2

2.6. *Lenguajes de programación utilizados*

Para el mapeado de datos en FUXA entre variables OPC y tópicos MQTT se ha utilizado el lenguaje javascript.

En la parte de control, se ha empleado el software MATLAB para grabar los datos de los ensayos en bucle abierto para caracterizar los dos sistemas de caudal. También se ha utilizado posteriormente para estudiar los controladores sintonizados en bucle cerrado, antes de desarrollar los controladores en el contenedor Docker.

Para el desarrollo de los controladores en el contenedor Docker, se ha hecho uso de Python con los módulos `pyyaml` y `paho-mqtt` que permite la carga de archivos de configuración en formato `YAML` y la gestión del cliente MQTT, respectivamente.

3. Resultados

3.1. *SCADA de la planta MD en FUXA*

En la Figura 3(a) se muestra un detalle de la ventana principal de la planta MD, en la que se pueden consultar todas las variables monitorizadas y actuar de forma manual sobre los actuadores (bombas para el caudal y válvulas de tres vías para las temperaturas de agua de alimentación a los módulos). También se incluye un apartado de gráficas en el que se muestran de forma visual los registros de temperatura, caudal y nivel de los tanques de alimentación.

La Figura 3(b) expone la interfaz elaborada para los ensayos en bucle cerrado de los controladores integrados en la red Docker que se comunican con FUXA mediante tópicos MQTT. Se incluyen botones para cambiar entre el modo manual y el modo automático, entradas para los *setpoints* y puntos de operación manuales de las bombas, y un script para efectuar el test automatizado con los pasos presentados en la Tabla 1 de la Sección 2.5.

3.2. *Red de contenedores Docker*

El esquema de la red de contenedores se presenta en la Figura 4. La red se comunica de manera externa con el PLC de la planta mediante un servidor OPC-UA e, internamente, se utiliza el protocolo MQTT para la actuación de los controladores.

La red está formada por: (i) Contenedor con FUXA. (ii) Contenedor *MQTT Broker* utilizando la imagen `eclipse-mosquitto v2`, necesario para utilizar el protocolo MQTT. El *Broker* se encarga de centralizar los mensajes de los dispositivos publicadores, llamados *publishers*, y los envía a los dispositivos que estén suscritos a dicha ruta, llamados *subscribers*. (iii) Contenedor `md-control` basado en la imagen `python:3.13.3-slim`. Contiene el software elaborado en Python con los dos controladores PI sintonizados.

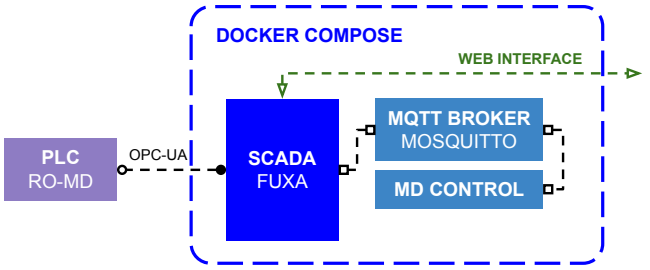
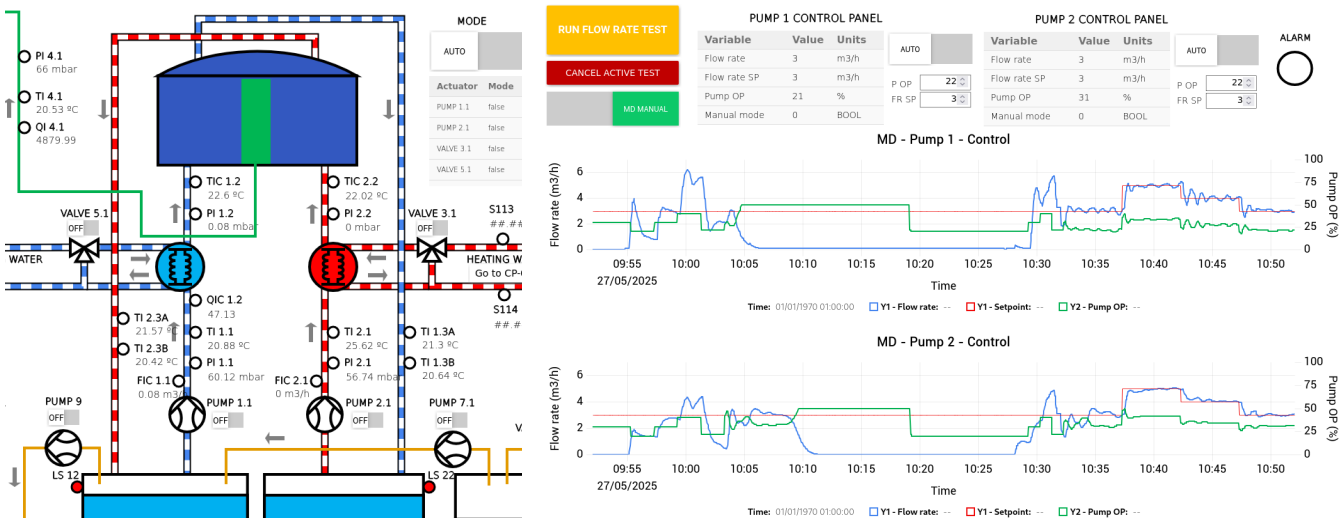


Figura 4: Esquema de la red de contenedores desarrollado con la herramienta `docker-compose` para el SCADA de la planta MD.

El software se encuentra disponible con acceso público en el repositorio de GitHub: `https://github.com/AaronPB/MD-controller`, en el que también se incluye el archivo de configuración del sistema `docker-compose` utilizado.

3.3. *Control del caudal de la planta MD*

En la Tabla 2 se muestran los valores promedios obtenidos de las funciones $G(s)$ de la planta MD, siguiendo el procedimiento en bucle abierto descrito en la Sección 2.5.



(a) Detalle del monitorizado de la instrumentación de la planta y el estado de los actuadores.

(b) Panel para gestionar los controladores diseñados vía la red Docker.

Figura 3: Interfaces del SCADA desarrollado en FUXA para la planta MD.

Tabla 2: Funciones de transferencia $G(s)$ de los caudales del flujo frío, FFR_1 y del flujo caliente, FFR_2. $P_{1\text{-}OP}$ y $P_{2\text{-}OP}$ corresponden a los porcentajes d funcionamiento de las respectivas bombas.

$G(s)$	$Y(s)$	$U(s)$	k [m^3/h %]	τ [s]	θ [s]
$G_1(s)$	FFR_1	$P_{1\text{-}OP}$	0.150	25.43	8.28
$G_2(s)$	FFR_2	$P_{2\text{-}OP}$	0.132	46.57	13.71

Con los modelos dinámicos estimados se han sintonizad controladores de tipo PI empleando el método SIMC (Sko gestad and Grimholt, 2012). En la Tabla 3 se presentan lo parámetros de los controladores PI, así como la ganancia del mecanismo *anti-windup*, K_{aw}, como $1/\sqrt{T_i}$. Se ha impuesto una constante de tiempo en bucle cerrado del 80 % de la constante de tiempo en el lazo abierto.

Tabla 3: Funciones de transferencia $C(s)$ de los controladores del flujo frí $C_1(s)$ y el flujo caliente $C_2(s)$ empleando el método SIMC. También se inclu yen las ganancias *anti-windup* K_{aw} seleccionadas.

$C(s)$	K_p [% h/m^3]	T_i [s]	K_{aw} [1/s]
$C_1(s)$	6.56	28.19	0.19
$C_2(s)$	7.60	51.14	0.14

En las Figuras 5 y 6 se muestran los resultados de los bu cles de control de caudal utilizando respectivamente un script en MATLAB y el contenedor Docker vía el panel del SCADA, mostrado en la Figura 3(b). Se ha seguido en ambos casos el procedimiento detallado en la Tabla 1 de la Sección 2.5.

Se pueden observar ciertas diferencias entre los resultados de ambos ensayos, sobre todo en los caudales de la bomba 1.1. Estas diferencias pueden deberse principalmente a cómo se han integrado los controladores en MATLAB y en el contenedor Docker en Python, y las etapas en las que se va procesando la información.

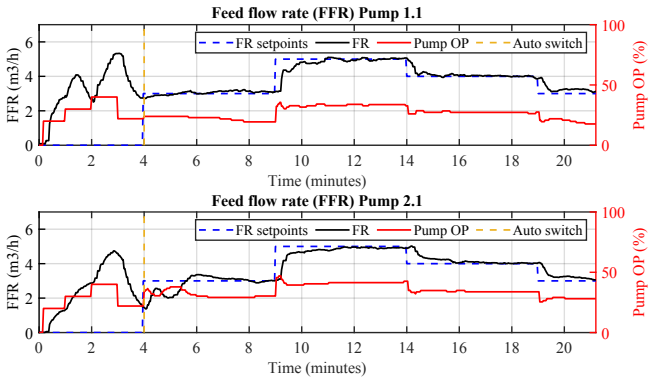


Figura 5: Resultados del arranque y control de los caudales de la planta MD utilizando MATLAB.

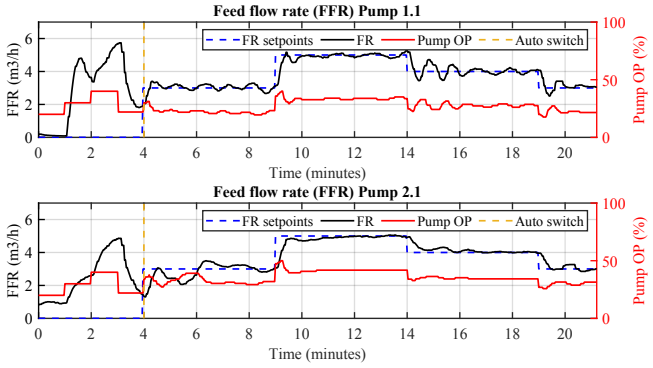


Figura 6: Resultados del arranque y control de los caudales de la planta MD utilizando FUXA y la red Docker.

En el caso de MATLAB, el control se realiza en un solo bucle que se va procesando cada 2 segundos. En dicho bucle, se reciben los valores del servidor OPC, se determinan los valores de salida de los controladores y, si se supera la banda de histéresis del 1 % con respecto al último valor enviado, se escriben los nuevos puntos de operación de las bombas.

En el caso de la red Docker, se tiene un script previo en

el que se mapean las variables de entrada y salida del servidor OPC y los tópicos MQTT. Posteriormente, estos tópicos son consultados en el bucle interno del contenedor, que sigue el mismo proceso que el comentado en MATLAB. Tanto el script como el bucle de los controladores del contenedor Docker se ejecutan cada 2 segundos, pero en paralelo y de manera independiente. Esto puede dar lugar a un desajuste de dos muestras en los datos y, por tanto, un retardo adicional que no se ha considerado al realizar las pruebas en bucle abierto desde MATLAB.

También se puede ver una tendencia parecida con los caudales de la bomba 2.1, aunque no tan notoria como en el caso de la bomba 1.1. Esto se debe a las características de los sistemas. Como se puede observar en los parámetros obtenidos en la Tabla 2, la constante de tiempo del caudal de la bomba 2.1 es de casi el doble que el del caudal de la bomba 1.1.

Es posible solventar el inconveniente que presenta la regulación del caudal de la bomba 1.1 en la red Docker incrementando el retardo para compensar el desajuste de dos muestras que se puede producir, o bien eliminando el bucle interno del contenedor de los controladores, para que el proceso se realice directamente cuando se recibe un cambio en la salida. De esta manera, los controladores están sincronizados directamente con el script que mapea las variables OPC a MQTT en FUXA.

4. Conclusiones y trabajos futuros

En este trabajo se presenta el desarrollo del SCADA de una planta MD a escala demostrativa, dentro de un sistema aislado y modular que permite su escalabilidad y transferencia a otros entornos. Dentro de la red se incluye el control regulatorio de los caudales de la planta MD utilizando estrategias de control convencionales.

En el futuro se pretende evaluar otras estrategias de control para mejorar el control regulatorio con la red Docker implementada. Además, se quiere incluir el control de temperatura mediante la actuación de las válvulas de los circuitos frío y caliente, y un control jerárquico de mayor nivel en el que se gestione el arranque automático del sistema y se utilicen parámetros de consumo para aplicar técnicas de optimización sobre la planta completa. Por último, se propone mejorar la gestión de datos integrando en la red de contenedores una base de datos robusta como InfluxDB, debido a que FUXA gestiona una base de datos local con SQLite. Con ello se facilita la gestión, el exportado y la consulta de datos, vía FUXA o mediante *dashboards* más específicos, como Grafana (Grafana Labs, 2025).

Agradecimientos

Esta publicación es parte del proyecto de I+D+i PID2023-150739OB-I00, financiado/a por MCIN/ AEI/10.13039/501100011033/ y "FEDER Una manera de hacer Europa". El trabajo también ha sido financiado por el Proyecto de Fortalecimiento de Grupos de Investigación (código P_FORT_GRUPOS_2023/14 - Universidad de Almería), financiado por la Secretaría General de Investigación e Innovación de la Consejería de Universidad, Investigación e Innovación de la Junta de Andalucía, en el marco del Programa Operativo FEDER Andalucía 2021-2027; y parcialmente por el Programa LIFE de la Unión Europea, bajo el acuerdo de subvención número LIFE23-ENV-ES-SALTEAU/101148475.

Referencias

Ahmad, N., Baddour, R. E., 2014. A review of sources, effects, disposal methods, and regulations of brine into marine environments. Ocean & coastal management 87, 1–7.
DOI: 10.1016/j.ocecoaman.2013.10.020

Alkhudhiri, A., Darwish, N., Hilal, N., 2012. Membrane distillation: A comprehensive review. Desalination 287, 2–18.
DOI: 10.1016/j.desal.2011.08.027

Andrés-Mañas, J., Gil, J. D., Sánchez-Molina, J., Berenguel, M., Zaragoza, G., 2025. Operation, control and assessment of a full-scale membrane distillation unit for treating desalination brine in the context of greenhouse production. Journal of Cleaner Production 503, 145186.
DOI: 10.1016/j.jclepro.2025.145186

Babayigit, B., Abubaker, M., 2023. Industrial internet of things: A review of improvements over traditional SCADA systems for industrial automation. IEEE Systems Journal 18 (1), 120–133.
DOI: 10.1109/JSYST.2023.3270000

Gil, J. D., González, R. A., Sánchez-Molina, J., Berenguel, M., Rodríguez, F., 2024. Reverse osmosis desalination for greenhouse irrigation: Experimental characterization and economic evaluation based on energy hubs. Desalination 574, 117281.
DOI: 10.1016/j.desal.2023.117281

Gil, J. D., Roca, L., Ruiz-Aguirre, A., Zaragoza, G., Berenguel, M., 2018. Optimal operation of a solar membrane distillation pilot plant via nonlinear model predictive control. Computers & Chemical Engineering 109, 151–165.
DOI: 10.1016/j.compchemeng.2017.11.019

Grafana Labs, 2025. Grafana: The open observability platform. https://grafana.com/, Último acceso: 22 de mayo de 2025.

Ibrahim, M. H., Sayagh, M., Hassan, A. E., 2021. A study of how docker compose is used to compose multi-component systems. Empirical Software Engineering 26, 1–27.
DOI: 10.1007/s10664-020-09873-2

Lahti, J. P., Shamsuzzoha, A., Kankaanpää, T., 2011. Web-based technologies in power plant automation and SCADA systems: A review and evaluation. In: 2011 IEEE International Conference on Control System, Computing and Engineering. IEEE, pp. 279–284.
DOI: 10.1109/ICCSCE.2011.6190491

Martinetti, C. R., Childress, A. E., Cath, T. Y., 2009. High recovery of concentrated ro brines using forward osmosis and membrane distillation. Journal of membrane science 331 (1-2), 31–39.
DOI: 10.1016/j.memsci.2009.01.003

Phuyal, S., Bista, D., Izykowski, J., Bista, R., 2020. Design and implementation of cost efficient scada system for industrial automation. International Journal of Engineering and Manufacturing 10 (2), 15.
DOI: 10.5815/ijem.2020.02.02

Qasim, M., Badrelzaman, M., Darwish, N. N., Darwish, N. A., Hilal, N., 2019. Reverse osmosis desalination: A state-of-the-art review. Desalination 459, 59–104.
DOI: 10.1016/j.desal.2019.02.008

Sánchez, P. S., Vergel, J. D. G., Mañas, J. A. A., Zaragoza, G., Molina, J. A. S., Berenguel, M., 2024. Puesta en funcionamiento, control regulatorio y modelado de métricas de rendimiento de un sistema de destilación por membranas a escala comercial. Jornadas de Automática (45).
DOI: 10.17979/ja-cea.2024.45.10776

Shalaby, S., Kabeel, A., Abosheiasha, H., Elfakharany, M., El-Bialy, E., Shama, A., Vidic, R. D., 2022. Membrane distillation driven by solar energy: A review. Journal of Cleaner Production 366, 132949.
DOI: 10.1016/j.jclepro.2022.132949

Skogestad, S., Grimholt, C., 2012. The SIMC method for smooth PID controller tuning. PID control in the third millennium: Lessons learned and new approaches, 147–175.
DOI: 10.1007/978-1-4471-2425-2_5

Sollfrank, M., Loch, F., Denteneer, S., Vogel-Heuser, B., 2020. Evaluating docker for lightweight virtualization of distributed and time-sensitive applications in industrial automation. IEEE Transactions on Industrial Informatics 17 (5), 3566–3576.
DOI: 10.1109/TII.2020.3019907

Viader, G., Casal, O., Lefevre, B., de Arespacochaga, N., Echevarría, C., López, J., Valderrama, C., Cortina, J., 2021. Integration of membrane distillation as volume reduction technology for in-land desalination brines management: Pre-treatments and scaling limitations. Journal of environmental management 289, 112549.
DOI: 10.1016/j.jenvman.2021.112549