Máster Universitario de Investigación en Ingeniería de Software y Sistemas Informáticos

Ingeniería de Software – 31105128

Detección inteligente de errores en la industria 4.0 a través de un Sistema IoT: Control de cuellos de botella producidos en los procesos de envasado

Trabajo fin de Máster

Autor: Manuel José Rodríguez Aguilar

Director: Ismael Abad Cardiel

Curso Académico 2017/2018

Convocatoria de Febrero

Máster Universitario de Investigación en Ingeniería de Software y Sistemas Informáticos

Ingeniería de Software - 31105128

Detección inteligente de errores en la industrial 4.0 a través de un Sistema IoT: Control de cuellos de botella producidos en los procesos de envasado.

Tipo de trabajo: B

Autor: Manuel José Rodríguez Aguilar

Director: Ismael Abad Cardiel

CALIFICACIONES

DECLARACIÓN JURADA DE AUTORÍA DEL TRABAJO CIENTÍFICO, PARA LA DEFENSA DEL TRABAJO FIN DE MÁSTER

Fecha: 10/02/2018

Quién suscribe:

Autor(a): Manuel José Rodríguez Aguilar
D.N.I/N.I.E/Pasaporte.: 46684778E

Hace constar que es la autor(a) del trabajo:

Detección inteligente de errores en la industria 4.0 a través de un Sistema IoT: Control de cuellos de botella producidos en los procesos de envasado

En tal sentido, manifiesto la originalidad de la conceptualización del trabajo, interpretación de datos y la elaboración de las conclusiones, dejando establecido que aquellos aportes intelectuales de otros autores se han referenciado debidamente en el texto de dicho trabajo.

DECLARACIÓN:

- ✓ Garantizo que el trabajo que remito es un documento original y no ha sido publicado, total ni parcialmente por otros autores, en soporte papel ni en formato digital.
- ✓ Certifico que he contribuido directamente al contenido intelectual de este manuscrito, a la génesis y análisis de sus datos, por lo cual estoy en condiciones de hacerme públicamente responsable de él.
- ✓ No he incurrido en fraude científico, plagio o vicios de autoría; en caso contrario, aceptaré las medidas disciplinarias sancionadoras que correspondan.

Fdo. Manuel José Rodríguez Aguilar

Impreso TFdM05_Autor. Autorización de publicación

y difusión del TFdM para fines académicos

Autorización

Autorizo/amos a la Universidad Nacional de Educación a Distancia a difundir y utilizar, con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la memoria de este Trabajo Fin de Máster, como el código, la documentación y/o el prototipo desarrollado.

Firma del/los Autor/es

Juan del Rosal, 16
28040, Madrid

Tel: 91 398 89 10
Fax: 91 398 89 09
www.issi.uned.es

RESUMEN

Este trabajo trata de resolver la problemática existente en líneas de envasado a la hora de detectar cuellos de botella. Un cuello de botella en un proceso productivo es una fase de la cadena de producción más lenta que otras, que ralentiza el proceso de producción global.

Este tipo de problemática es en muchas ocasiones indetectable. Según expertos, puede ser dado por factores humanos, de vigilancia, carencias productivas, falta de mantenimiento, etc.

Es por esta razón, que el sistema que se diseñará aportará funcionalidad a partir del análisis previo de estos factores que permita detectar este tipo de problemáticas y aprovechar los datos que puede proveer una línea de envasado automatizado para crear un ecosistema basado en el paradigma IoT y concretamente, para alcanzar un sistema dotado para la industria 4.0.

Como antecedente, los sistemas de control de las estaciones de trabajo en una línea de envasado están normalmente centrados en el proceso secuencial para cumplir un objetivo empresarial sobre un producto determinado. Tal y como se verá en el contexto de investigación, existen aspectos con los cuales se puede determinar cuándo puede existir un cuello de botella y, en este sentido, el control de una estación de trabajo tiene un alcance limitado para cubrir la problemática, con lo cual, no se puede determinar de forma automática cuando se produce un cuello de botella.

El presente trabajo, proveerá una iniciativa centrada en el diseño de sensores y funciones para recopilar datos necesarios y dotar de una detección inteligente de cuellos de botella tal que permita recopilar y almacenar información con el objetivo de monitorizar datos clave en la detección del cuello de botella,

generar reportes para tomar decisiones ante los riesgos detectados y revisar históricos para crear comparativas con los datos y utilizar los antecedentes para advertir problemáticas.

A partir de aquí, en un contexto real, el sistema trabajaría como un soporte a las operaciones dentro de la línea de envasado. Ofreciendo valor al negocio al anticiparse a problemáticas derivadas de factores que provoquen cuellos de botella.

ABSTRACT

This work tries to solve the existent problematic in lines of packaging at the time of detecting bottlenecks. A bottleneck in a production process is one phase of the production chain slower than others, which slows down the overall production process.

This type of problem is often undetectable. According to experts, it can be given by human factors, surveillance, lack of production, lack of maintenance, etc.

It is for this reason that the system to be designed will provide functionality from the previous analysis of these factors to detect this type of problems and take advantage of the data that can be provided by an automated packaging line to create an ecosystem based on the IoT paradigm and specifically, to achieve a gifted system for industry 4.0.

As background, the control systems of work stations in a packaging line are usually focused on the sequential process to meet a business objective on a given product. As will be seen in the research context, there are aspects with which one can determine when a bottleneck may exist and, in this sense, the control of a work station has a limited scope to cover the problem, with which, cannot be determined automatically when a bottleneck occurs.

The present work will provide an initiative focused on the design of sensors and functions to collect necessary data and provide an intelligent detection of bottlenecks such that it can collect and store information to monitor key data in the detection of the bottleneck, generate reports to make decisions against the detected risks and review historical data to create comparisons with the data and use the background to warn of problems.

From here, in a real context, the system would work as a support to the operations within the packaging line. Offering value to the business by anticipating problems derived from factors that cause bottlenecks.

PALABRAS CLAVE

Línea de envasado, estación de trabajo, cuellos de botella, industria 4.0, IoT, sensor de cadencia, RFID, sensor de envases, maqueta web, Arduino Yun, Autómata programable, proxy, Spring Framework, Middleware, contenedor Tomcat, históricos, reporte online, Bootstrat

CONTENIDO

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

1. INTRODUCCIÓN

1.1. JUSTIFICACIÓN

Dada la dificultad para poder evaluar un problema de cuello de botella generado por factores diversos como el humano, falta de mantenimiento y vigilancia entre otros. Se ha visto como una oportunidad crear un trabajo de investigación que presente el contexto actual de este tipo de problemáticas y mostrar una solución que aporte valor para que se puedan tomar decisiones ante una serie de datos analizados y monitorizados.

La iniciativa se prevé como un marco de referencia inicial para poder anticiparse a este tipo de problemas.

Para conseguir este marco, se diseñarán tres funcionalidades base dentro de un contexto de sistema con un diseño arquitectónico tanto en infraestructura y sistemas como en software.

Las tres funcionalidades se basarán en la monitorización, el análisis y el reporte de datos, esto se justifica porqué son principios indicados para los expertos para poder anticiparse a errores de cuello de botella:

- Medir las variables de rendimiento clave en un cuello de botella.
- gestionar y analizar de forma continua los cuellos de botella, monitorizarlos.

1.2. OBJETIVOS

El objetivo general del trabajo ha sido investigar el impacto de las problemáticas generadas por los cuellos de botella en líneas de envasado de productos embotellados y mostrar una iniciativa de diseño basada en el paradigma IoT y la Ingeniería del Software para poder solventarlo.

Para poder dar valor a este objetivo general, se han ideado una serie de objetivos individuales los cuales se han transmitido a través de las siguientes actividades:

- Estudio individual para identificar las características y factores que provocan este tipo de problemática.
- Presentar un estado del arte del contexto de línea de envasado con su composición a partir de sus estaciones de trabajo.
- Investigar los componentes clave de cada estación y preparar un cuadro con las causas y consecuencias de los problemas de cuello de botella para cada una de las estaciones.
- Presentar una propuesta para mejorar la situación a partir de una idea funcional a partir de un ciclo compuesto por tres fases:
    - Control y tratamiento de sensores
    - Análisis de los datos adquiridos
    - Decisiones y cambios sobre la línea analizada.
- Presentar un modelo de Sistema IoT en bloques que modele la problemática.
- Solución propuesta a partir de una análisis funcional, arquitectónico y técnico para finalmente generar un prototipo a través de tres casos de uso que demuestre cómo funcionará el sistema a alto nivel con datos simulados.
- Conclusiones y línea de trabajo futuros con el output proporcionado en la presente investigación.

1.3. BREVE RESUMEN DE APARTADOS

En el capítulo 2, se mostrará el resultado efectuado en la investigación analizada, principalmente se verá el estado de arte del problema investigado, el contexto de un proceso de envasado con sus estaciones principales, causas y consecuencias de la problemática analizada y la solución propuesta para mejorar la situación.

El capítulo 3, se centra en la especificación y diseño de la solución IoT ideada para resolver la problemática analizada, en concreto el capítulo se centra en los siguientes apartados:

- Especificación funcional. Se mostrará la iniciativa de funcionalidades ideadas para el sistema y pantallas maqueta de los casos de uso de monitorización, gestión de reporte e históricos.
- Especificación técnica. En el subcapítulo se muestran los diferentes componentes técnicos y arquitectónicos necesarios para cubrir las funcionalidades diseñadas.
- Implementación de solución. Se presenta el prototipo del sistema. Pila de componentes tecnológicos integrados y estructura de código del sistema.

El capítulo 4 mostrará la evaluación con los resultados obtenidos de las funcionalidades aplicadas al prototipo en forma de interfaz de usuarios, en el mismo se justifican los componentes utilizados para mostrar la información.

El capítulo 5 se centrará en realizar una comparativa del sistema diseñado con otras soluciones mostrando además como podría encajar de forma híbrida con otros diseños.

Por último, los dos capítulos finales 6 y 7 mostrarán una conclusión a la investigación y unas líneas de trabajo futuro respectivamente.

1.4. ALCANCE

Como alcance, se establece la investigación de la problemática presentada sobre cuellos de botella en líneas de envasado. La presentación de una solución basada en el paradigma IoT en base a la industria 4.0 y el diseño de esta iniciativa basada en la Ingeniería del Software teniendo como base las disciplinas estudiadas de computación ubicua, generación automática de código y el paradigma de la orientación a servicios.

Dentro del alcance del presente trabajo, se cubre un prototipo basado en las tres funcionalidades principales del sistema: Monitorización de variables clave, Gestión de históricos y reporte de informes. A partir del diseño de estas funcionalidades, se implementa un prototipo que constará de tres interfaces de usuario con datos de tipo test simulados para mostrar cómo funcionará el producto diseñado que mejorará la problemática investigada.

1.5. RESULTADO OBTENIDO

Después de finalizar el proceso de investigación se elaboraron una serie de conclusiones con las que se propuso una solución basada en el paradigma IoT para mejorar la problemática planteada sobre los cuellos de botella en líneas de envasado.

A partir de aquí, se ha obtenido como resultado:

- Memoria con el resultado de la investigación sobre la problemática analizada con sus conclusiones.
- Diseño de Software que cubre el análisis funcional, arquitectónico y técnico de la propuesta para solucionar la problemática analizada.

- Prototipo del Software diseñado con pruebas de funcionalidades de Monitorización de estación, reporte diario de datos y gráfico obtenido de datos históricos.

A partir del diseño de Software se ha generado la estructura de un proyecto siguiendo las directrices Java Web diseñadas en la arquitectura y se ha implementado la base del código que cubrirá las funcionalidades diseñadas.

Una vez creada esta estructura, se ha implementado un prototipo con las tres principales interfaces de usuario del proyecto: Monitorización de estación, reporte de datos y visualización de gráficos con históricos de datos.

Al finalizar el prototipo se evaluó a partir de la realización de una serie de las cuales finalmente han permitido tener una visibilidad a alto nivel del funcionamiento del sistema a partir del uso de datos simulados en una clase Test dentro del sistema.

La implementación del prototipo se podrá ver en el capítulo 3 apartado 3.3. Posteriormente, los resultados de estas pruebas se mostrarán con mayor detalle en el capítulo 4, que es específico de evaluación de este documento.

1.6. PLANIFICACIÓN

1.6.1. PLANIFICACIÓN BASE

	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin	Pr	Apuntes
1		**Proyecto TFM**	**90 días?**	**lun 10/07/17**	**vie 10/11/17**		**Jornadas 3 horas**
2		Elaboración Plan de trabajo	1 día	lun 10/07/17	lun 10/07/17		
3		Proceso de investigación	41 días?	mar 11/07/17	mar 05/09/17		
4		**Investigar viabilidad técnica Arduino y tecnologías Web**	**7 días**	**mar 11/07/17**	**mié 19/07/17**	**2**	
5		Prueba de concepto con parte Arduino	2 días	mar 11/07/17	mié 12/07/17		
6		Prueba de concepto con parte Web	3 días	jue 13/07/17	lun 17/07/17	5	
7		Pruebas de conectividad componentes	2 días	mar 18/07/17	mié 19/07/17	6	
8		Definir estrategia de simulación	3 días	jue 20/07/17	lun 24/07/17	7	
9		Recopilación de información contexto problema	5 días	mar 25/07/17	lun 31/07/17	8	
10		Investigar información recopilada contexto problema	4 días	mar 01/08/17	vie 04/08/17	9	
11		**Documentar Investigación**	22 días?	lun 07/08/17	mar 05/09/17	10	
12		Definición Problema Abordado	1 día?	lun 07/08/17	lun 07/08/17		
13		Contexto del problema abordado	2 días	mar 08/08/17	mié 09/08/17	12	
14		Solución propuesta para mejorar situación	3 días	jue 10/08/17	lun 14/08/17	13	Descanso 14/08-27/08
15		**Proyección de la solución propuesta**	**7 días**	**lun 28/08/17**	**mar 05/09/17**		
16		Memoria de proyección	3 días	lun 28/08/17	mié 30/08/17		
17		Relación con CU	2 días	jue 31/08/17	vie 01/09/17	16	
18		Relación con GAC	2 días	lun 04/09/17	mar 05/09/17	17	
19		Especificación y diseño de la solución-prototipo IoT de mejora	44 días?	mié 06/09/17	lun 06/11/17	18	
20		Especificar arquitectura software base Arduino Industrial	2 días	mié 06/09/17	jue 07/09/17		
21		Especificar arquitectura software base Aplicación Web	3 días	vie 08/09/17	mar 12/09/17	20	
22		Definición requisitos y funcional	3 días	mié 13/09/17	vie 15/09/17	21	
23		Defincón especificación técnica	3 días	lun 18/09/17	mié 20/09/17	22	
24		Definición datos de simulación	2 días	jue 21/09/17	vie 22/09/17	23	
25		Definir forma de evaluar prototipo	1 día	lun 25/09/17	lun 25/09/17	24	
26		**Implementación**	**15 días**	**mar 26/09/17**	**lun 16/10/17**	**25**	
27		Implementación arduino industrial	3 días	mar 26/09/17	jue 28/09/17		
28		Implementación aplicación WEB	9 días	vie 29/09/17	mié 11/10/17	27	
29		Fase de pruebas y evaluación	3 días	jue 12/10/17	lun 16/10/17	28	
30		**Documentar especificación e implementación**	**15 días?**	**mar 17/10/17**	**lun 06/11/17**	**29**	
31		Definición arquitectura	1 día?	mar 17/10/17	mar 17/10/17		
32		**Componentes de arquitectura de infraestructuras y sistemas**	**3 días?**	**mié 18/10/17**	**vie 20/10/17**	**31**	
33		Arquitectura de sistemas e infraestructuras	1 día?	mié 18/10/17	mié 18/10/17		
34		Dibujar Componentes de simulación (Revisar si inclusión)	2 días	jue 19/10/17	vie 20/10/17	33	
35		**Componentes de arquitectura de Software**	**1 día?**	**lun 23/10/17**	**lun 23/10/17**	**34**	
36		Arquitectura de componentes Software	1 día?	lun 23/10/17	lun 23/10/17		
37		**Diseño técnico**	**4 días?**	**mar 24/10/17**	**vie 27/10/17**	**36**	
38		Modelo de datos	1 día?	mar 24/10/17	mar 24/10/17		
39		Componentes y Clases que intervienen	1 día?	mié 25/10/17	mié 25/10/17	38	
40		Diagrama de secuencia principal	2 días	jue 26/10/17	vie 27/10/17	39	
41		**Documentación implementación de la solución**	**2 días?**	**lun 30/10/17**	**mar 31/10/17**	**40**	
42		Recortes código destacables	1 día?	lun 30/10/17	lun 30/10/17		
43		Pila de componentes integrados	1 día?	mar 31/10/17	mar 31/10/17	42	
44		Documentar evaluación del producto	3 días	mié 01/11/17	vie 03/11/17	43	
45		Revisión documentación	1 día	lun 06/11/17	lun 06/11/17	44	
46		Documentar resto memoria y finalizar	4 días?	mar 07/11/17	vie 10/11/17		
47		Documentar Comparativa con otras soluciones	1 día?	mar 07/11/17	mar 07/11/17	45	
48		Documentar resto de memoria (Intro, agradec., objeto, concl., etc)	1 día?	mié 08/11/17	mié 08/11/17	47	
49		Revisar formato de memoria	1 día?	jue 09/11/17	jue 09/11/17	48	
50		Entrega de TFM	1 día	vie 10/11/17	vie 10/11/17	49	

1.6.2. ACTUALIZACIÓN PLANIFICACIÓN

		Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin	Pre	% comple
1			**Proyecto TFM**	**134 días?**	**lun 10/07/17**	**jue 11/01/18**		**76%**
2	✓		Elaboración Plan de trabajo	1 día	lun 10/07/17	lun 10/07/17		100%
3	✓		Proceso de investigación	41 días?	mar 11/07/17	mar 05/09/17		100%
4	✓		**Investigar viabilidad técnica Arduino y tecnologías Web**	**7 días**	**mar 11/07/17**	**mié 19/07/17**	**2**	100%
5	✓		Prueba de concepto con parte Arduino	2 días	mar 11/07/17	mié 12/07/17		100%
6	✓		Prueba de concepto con parte Web	3 días	jue 13/07/17	lun 17/07/17	5	100%
7	✓		Pruebas de conectividad componentes	2 días	mar 18/07/17	mié 19/07/17	6	100%
8	✓		Definir estrategia de simulación	3 días	jue 20/07/17	lun 24/07/17	7	100%
9	✓		Recopilación de información contexto problema	5 días	mar 25/07/17	lun 31/07/17	8	100%
10	✓		Investigar información recopilada contexto problema	4 días	mar 01/08/17	vie 04/08/17	9	100%
11	✓		**Documentar Investigación**	22 días?	lun 07/08/17	mar 05/09/17	10	100%
12	✓		Definición Problema Abordado	1 día?	lun 07/08/17	lun 07/08/17		100%
13	✓		Contexto del problema abordado	2 días	mar 08/08/17	mié 09/08/17	12	100%
14	✓		Solución propuesta para mejorar situación	3 días	jue 10/08/17	lun 14/08/17	13	100%
15	✓		**Proyección de la solución propuesta**	**7 días**	**lun 28/08/17**	**mar 05/09/17**		**100%**
16	✓		Memoria de proyección	3 días	lun 28/08/17	mié 30/08/17		100%
17	✓		Relación con CU	2 días	jue 31/08/17	vie 01/09/17	16	100%
18	✓		Relación con GAC	2 días	lun 04/09/17	mar 05/09/17	17	100%
19	✓		Especificación y diseño de la solución-prototipo IoT de mejora	61 días	mié 06/09/17	mié 29/11/17	18	**100%**
20	✓		Especificar arquitectura software base Arduino Industrial	2 días	mié 06/09/17	jue 07/09/17		100%
21	✓		Especificar arquitectura software base Aplicación Web	3 días	vie 08/09/17	mar 12/09/17	20	100%
22	✓		Definición requisitos y funcional	3 días	mié 13/09/17	vie 15/09/17	21	100%
23	✓		Definción especificación técnica	3 días	lun 18/09/17	mié 20/09/17	22	100%
24	✓		Definición datos de simulación	2 días	jue 21/09/17	vie 22/09/17	23	100%
25	✓		Definir forma de evaluar prototipo	1 día	lun 25/09/17	lun 25/09/17	24	100%
26	✓		**Especificación Funcional y técnica**	**45 días**	**jue 28/09/17**	**mié 29/11/17**		**100%**
27	✓		**Documentar funcionalidades**	**18 días**	**jue 28/09/17**	**lun 23/10/17**		**100%**
28	✓		Modelo global del sistema	3 días?	jue 28/09/17	lun 02/10/17		100%
29	✓		Fichas de funcionalidades	13 días	mar 03/10/17	jue 19/10/17		100%
30	✓		Modelo de dominio	2 días	vie 20/10/17	lun 23/10/17		100%
31	✓		**Componentes de arquitectura de infraestructuras y sistemas**	**10 días**	**mar 24/10/17**	**lun 06/11/17**	**30**	**100%**
32	✓		Arquitectura de sistemas e infraestructuras	5 días	mar 24/10/17	lun 30/10/17		100%
33	✓		Dibujar Componentes de simulación (Revisar si inclusión)	5 días	mar 31/10/17	lun 06/11/17	32	100%
34	✓		**Componentes de arquitectura de Software**	**7 días**	**vie 27/10/17**	**lun 06/11/17**	**33**	**100%**
35	✓		Arquitectura de componentes Software	6 días	vie 27/10/17	vie 03/11/17		100%
36	✓		**Diseño técnico**	**7 días**	**lun 30/10/17**	**mar 07/11/17**	**35**	**100%**
37	✓		Modelo de datos	2 días	lun 30/10/17	mar 31/10/17		100%
38	✓		Componentes y Clases que intervienen	3 días	mié 01/11/17	vie 03/11/17	37	100%
39	✓		Diagrama de secuencia principal	2 días	lun 06/11/17	mar 07/11/17	38	100%
40	✓		Documentar y revisar arquitectura y diseño técnico	15 días	mié 08/11/17	mar 28/11/17	39	100%
41			Implementación prototipo	**27 días**	**jue 30/11/17**	**vie 05/01/18**		**0%**
42			Implementación prototipo aplicación WEB	9 días	jue 30/11/17	mar 12/12/17		0%
43			Implementación prototipo arduino yun industrial	4 días	mié 13/12/17	lun 18/12/17	42	0%
44			Simulacion dependencias	4 días	mar 19/12/17	vie 22/12/17	43	0%
45			Fase de pruebas, resultados y evaluación	6 días	lun 25/12/17	lun 01/01/18	44	0%
46			**Documentación implementación de la solución**	**5 días**	**lun 01/01/18**	**vie 05/01/18**	**45**	**0%**
47			Recortes código destacables	1 día?	lun 01/01/18	lun 01/01/18		0%
48			Pila de componentes integrados	1 día?	mar 02/01/18	mar 02/01/18	47	0%
49			Documentar evaluación del producto	3 días	mié 03/01/18	vie 05/01/18	48	0%
50			Correcciones según indicaciones Director de TFM hasta finalización	42 días	jue 30/11/17	vie 26/01/18		0%
51			Documentar resto memoria y finalizar	4 días?	lun 08/01/18	jue 11/01/18	49	**0%**
52			Documentar Comparativa con otras soluciones	1 día?	lun 08/01/18	lun 08/01/18		0%
53			Documentar resto de memoria (Intro, agradec., objeto, concl., etc)	1 día?	mar 09/01/18	mar 09/01/18	52	0%
54			Revisar formato de memoria	1 día?	mié 10/01/18	mié 10/01/18	53	0%
55			Versión con todos los apartados TFM (Adelantar primera semana Enero)	1 día	jue 11/01/18	jue 11/01/18	54	0%

2. CONTEXTO DE INVESTIGACIÓN

2.1. INTRODUCCIÓN

Esta investigación se centra en el marco de los problemas relacionados con los cuellos de botella provocados en una línea de producción de envasado y como el paradigma de Internet de las cosas, en adelante IoT, puede ayudar a detectarlos.

No obstante, este tipo de problema no solo se limita a los problemas que pueden derivar a un problema productivo, el problema del cuello de botella se encuentra en muchos más ámbitos como los procesos económicos y logísticos o en procesos científicos aplicados a la biología, física y química, genética, etc.

ILUSTRACIÓN 1 CUELLO DE BOTELLA GENERADO POR UN PEAJE DE SERVICIO EN AUTOPISTA

En [SIM01] se describen siete ejemplos que pueden representar un cuello de botella, según el ámbito en el que impacte:

Aprobaciones de proyectos

Un proyecto solar tarda 2 meses en construir, pero espera 4 meses para la aprobación legal.

Fabricación

Una fase intermedia de un proceso de fabricación puede manejar 10 unidades por hora cuando las siguientes fases pueden manejar 100 unidades por hora.

Transporte

Una estación de tren puede crear un cuello de botella en horas punta según el flujo de viajeros (por ejemplo, una parada de gran interés turístico), si esta parada no está correctamente gestionada por personal que ayude a las personas a circular y salir de los trenes puede generar un cuello de botella.

Cuello de botella en una Red informática

Un enrutador WIFI lento conectado a una red eficiente y de alto ancho de banda.

Comunicación

Un desarrollador de software que pasa un promedio de 6 horas al día en reuniones y 2 horas al día de codificación.

Informática

Un sistema tiene un procesador rápido, memoria y disco duro, pero tiene un proceso de fondo innecesario en ejecución que está consumiendo grandes cantidades de recursos y ralentizar la máquina.

Procesos de negocios

Representantes de servicio al cliente en una línea aérea son que incapaces de manejar una variedad de situaciones sin obtener la aprobación de un

responsable. Los gerentes a menudo están ocupados, lo que significa que los representantes y los clientes a menudo se quedan esperando.

Según [WIK01], en el caso del proceso productivo, el problema hace referencia cuando una fase de la cadena de producción va más lenta que otras, que ralentiza el proceso de producción global.

En definitiva, se formará un cuello de botella cuando en una situación exista una fase que ralentice a un proceso global.

La idea en esta investigación es mostrar el estado del arte en los cuellos de botella, revisarlo sobre un contexto de una línea de envasado actual y como el paradigma IoT dentro de la nueva generación de la industria 4.0 puede impactar para ayudar a reducirlos a través de la gestión y monitorización de variables clave. A lo largo de este capítulo, se verán los detalles de cada uno de los aspectos de la investigación.

2.2. PROBLEMA INVESTIGADO: CUELLOS DE BOTELLA PRODUCIDOS EN LOS PROCESOS DE ENVASADO

Tal y como se ha indicado en la introducción, el problema del cuello de botella generado en un proceso productivo se genera cuando existe una fase que ralentiza al resto y globalmente no se obtiene la capacidad diseñada de producción.

Según la diferente bibliografía analizada, el origen del problema corresponde a una serie de factores como los que enumera [OPE01]:

1. Limitaciones de capacidad del equipo inherente.
2. Intercambios largos.
3. Problemas de confiabilidad mecánica.
4. Pérdidas de rendimiento.
5. Programación inapropiada, es decir, CCRs.

En la última causa se menciona CCR (traducido RRC), el autor lo describe como recursos de restricción de capacidad donde cualquier recurso si no se programa o administra correctamente, es probable que el flujo real del producto a través de la planta de producción se desvíe del flujo de producto planeado.

El mismo autor da un enfoque de las problemáticas que pueden generar cuellos de botella a través de las siguientes causas y aspectos:

1. La causa principal es generalmente la capacidad y el rendimiento del equipo, no la dotación de mano de obra.
2. Las causas principales incluyen pérdidas de rendimiento y problemas de confiabilidad, así como la capacidad inherente.
3. Los no cuellos de botella pueden convertirse en cuellos de botella debido a la variabilidad en los factores OEE (capacidad efectiva versus capacidad según placa informativa del fabricante en las máquinas que componen la línea productiva).
4. Las plantas funcionan a menudo alrededor del reloj, así que los cambios adicionales no son una solución factible.
5. Los cuellos de botella pueden moverse con la mezcla de productos.
6. Los cuellos de botella pueden no ser obvios - el inventario resultante y otros residuos se ocultan a menudo de la vista.

Una vez vistos las posibles causas y características en los problemas de los cuellos de botellas, [ATO01] muestra una serie de soluciones para desbloquear los cuellos de botellas en las líneas de producción:

- Mejorar los métodos de trabajo para reducir el cuello de botella (Mejora en producción y eficiencia).
- Mejorar el rendimiento en el cuello de botella.
- Garantizar que se reciba un trabajo de alta calidad en el cuello de botella.

- Colocar un buffer (almacenamiento) de seguridad delante de un cuello de botella para evitar quedarse sin trabajo.
- Reducir el tiempo de cambio.
- Realizar mantenimiento preventivo sobre las estaciones de trabajo.
- Añadir turnos, horas extraordinarias o trabajadores para reducir impacto.
- Adiestramiento del personal tal que un cuello de botella nunca esté inactivo debido a una ausencia.
- Medir las variables de rendimiento clave en un cuello de botella.

[PLA01], empresa especializada en soluciones para el control de cuellos de botella, además añade: "gestionar y analizar de forma continua los cuellos de botella, monitorizarlos".

Para finalizar, una de las teorías que ha estudiado las problemáticas de los cuellos de botella de manera formal es la de las restricciones (TOC) de Goldratt [WIK02], este físico basaba la teoría en dos principios:

- La teoría se basa en que los procesos de cualquier ámbito se mueven a la cadencia del paso más lento.
- La forma de equilibrar el proceso parte es utilizar un elemento acelerador en el paso más lento y lograr que trabaje hasta el alcance máximo de capacidad para poder acelerar el proceso completo.

En el artículo de [GES01] muestra el enfoque sistemático de esta teoría:

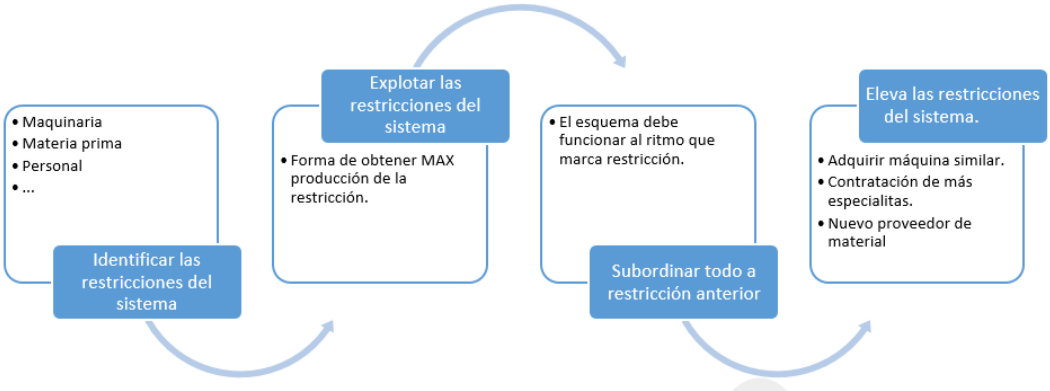


ILUSTRACIÓN 2: ENFOQUE SISTEMÁTICO EN CUIELLOS DE BOTELLA

Según el autor, si se eliminan las restricciones en algún paso previo a la última etapa se debe de volver a la primera fase para identificar las restricciones del sistema.

También se ha de tener en cuenta factores como el cambio de proveedor de elementos que componen la materia el producto al volver a analizar el primer paso.

En resumen, si contrastamos las soluciones que aportan los distintos autores se extrae que en común se debería de dotar de:

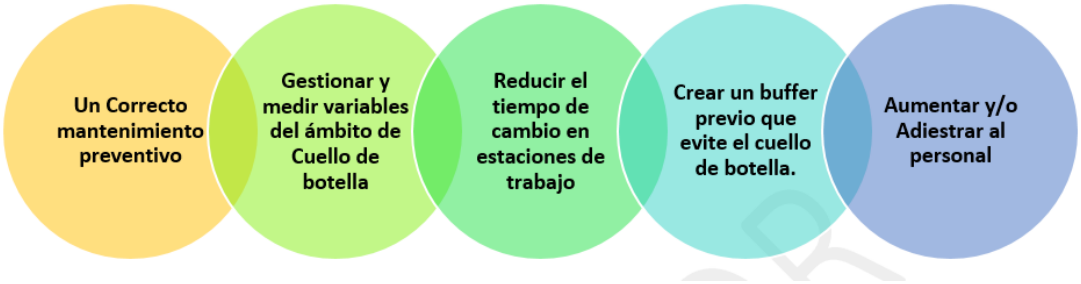


ILUSTRACIÓN 3: SOLUCIONES PARA DESBLOQUEAR LOS CUELLOS DE BOTELLA

2.3. CONTEXTO DE UNA LÍNEA DE ENVASADO AUTOMATIZADA

Una línea de envasado está formada por una serie de máquinas con una serie de objetivos concretos para completar un proceso productivo.

La línea de envasado puede estar compuesta por una sola máquina compacta, que cubra todo un proceso completo desde el envasado hasta el empaquetado del producto.

No obstante, por norma general un proceso de envasado está compuesto por varias estaciones de trabajo con maquinaría independiente que ejecutan diferentes fases en un proceso de envasado.

Según [WIK03]:

Packaging machines *are machines that complete stages of the packaging process. Examples include filling machines, sealing machines, wrapping machines, strapping machines, labelling machines and coding machines.*

Tal y como se puede ver en la expresión, el proceso de envasado puede estar cubierto por diferentes máquinas según su responsabilidad desde el llenado hasta el etiquetado y codificado.

Con objeto de trabajar dentro de un alcance específico dentro del conjunto de procesos posibles de envasado, se configura el estudio dentro de una línea de envasado a través de botellas de plástico compuesta por las siguientes estaciones de trabajo:

- Estación de distribución de envases.
- Estación de envasado.
- Estación de etiquetado
- Estación de empaquetado.
- Estación de paletizado.

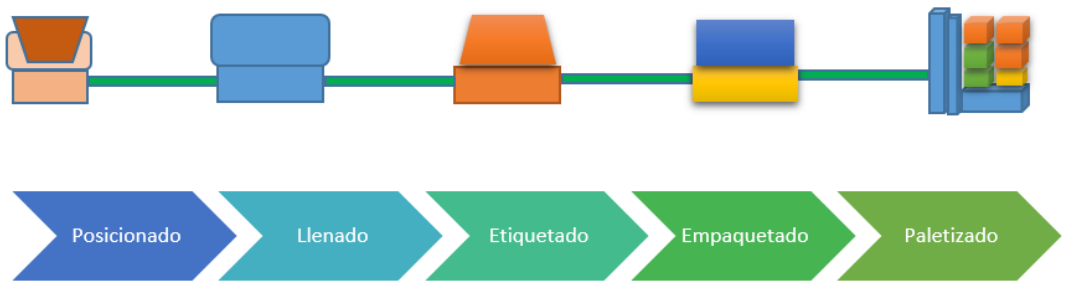


ILUSTRACIÓN 4: SECUENCIA DE LÍNEA DE ENVASADO

2.3.1. ESTACIONES DE TRABAJO: COMO SE COMPONE LA LÍNEA

Una vez vistas las estaciones de trabajo, en las siguientes líneas se muestran distintas formas de enlazarlas según las necesidades de cada producto a fabricar.

Para poder diseñar una línea de envasado, existen muchos procesos productivos y todos dependen de muchos factores además de la forma de fabricar el producto, en el caso de los envases de plástico, ocurre lo mismo, existen muchos tipos de envase en tamaño, forma, contenido, etc.

En las siguientes líneas, se muestran tres tipos de línea de envasado más común.

Línea de envasado en proceso lineal

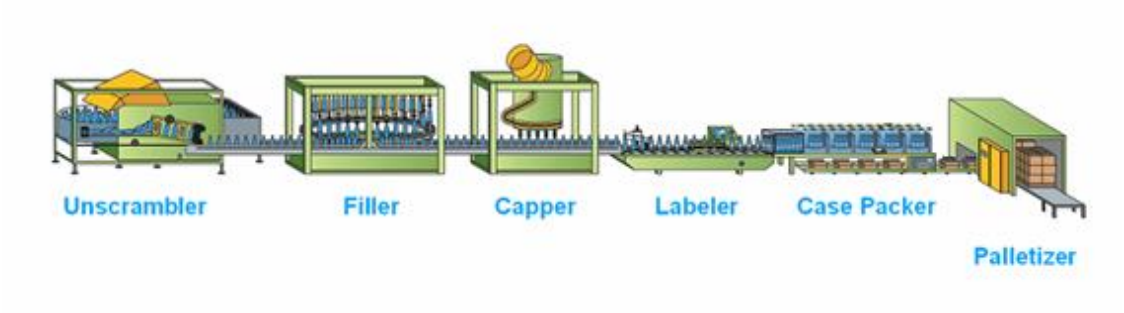


ILUSTRACIÓN 5: PROCESO LINEAL DE ENVASADO

La figura muestra las estaciones enlazadas de forma secuencial, tal y como se puede ver en este caso entre estaciones no existe mucho espacio, prácticamente la salida de una estación es la entrada de otra.

Para resolver la capacidad de absorción de envases entre estaciones y evitar cuellos de botella se utilizarán buffers de almacenamiento como la siguiente imagen:

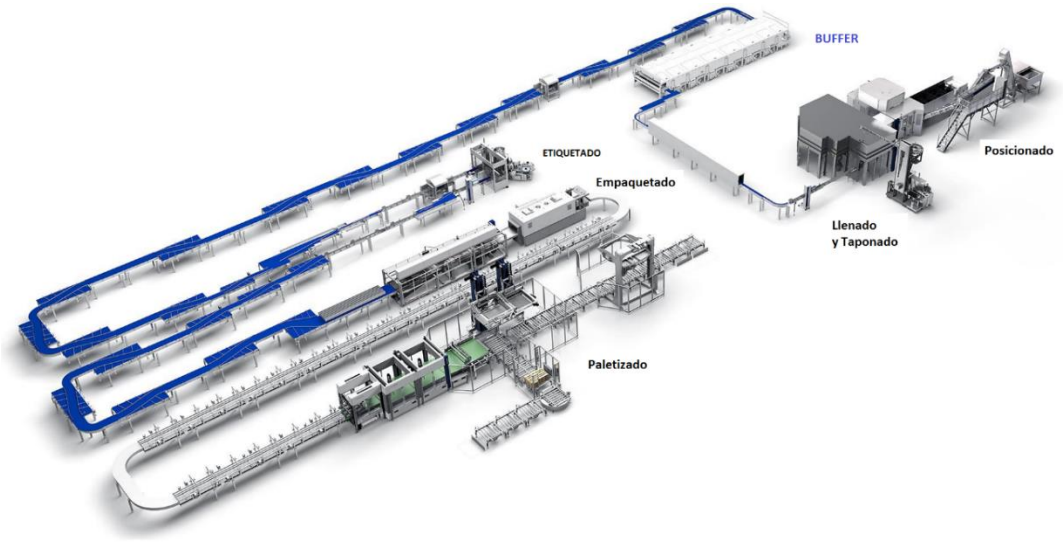


ILUSTRACIÓN 6: PROCESO DE ENVASADO DE GRAN ALMACENAJE DE TRANSPORTE

Línea de envasado en U:

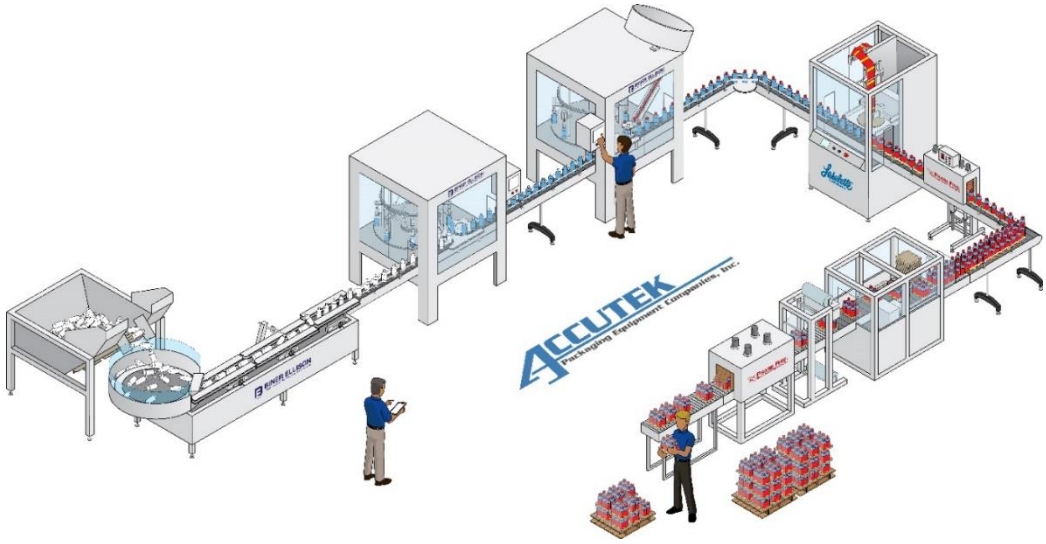


ILUSTRACIÓN 7: LINEA DE ENVASADO EN U

En esta simulación se puede observar una línea de envasado en forma de U, esta permite comprimir la línea en un espacio menor donde no se pueden instalar las estaciones en línea o con capacidad de largos transportes entre

Líneas de envasado en paralelo:

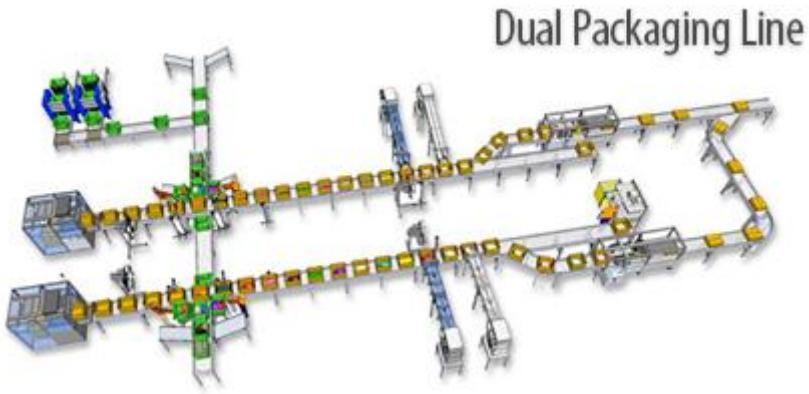


ILUSTRACIÓN 8: LINEA DE EMPAQUETADO DUAL

En los diseños de las líneas de envasado, puede existir la producción en paralelo de diversas líneas que finalmente se junten en la estación de paletizado.

2.3.2. GOBIERNO (AUTOMATIZACIÓN) EN LA LÍNEA DE ENVASADO

Por norma general y a no ser que las estaciones de trabajo estén integradas por un mismo fabricante, cada estación tendrá un control autónomo.

Esto quiere decir que cada una presentará un cuadro de mando y automatización propia.

Las máquinas podrán tener su propia estación mecánica, neumática e hidráulica de mando, aunque la energía la subministre el propietario de fábrica.

Partiendo de la base de máquinas totalmente automatizadas, tanto la potencia como el mando o maniobra eléctrico-electrónico suele venir preinstalado por el fabricante de máquina a partir de una automatización en mayor medida con Autómatas programables, microcontroladores y PC industrial.

Dentro de estos tres, el uso del autómata programable o PLC (Controlador lógico programable) [PLC01] tiene una gran extensión dentro de la automatización

industrial justificado por su gran adaptación, escalabilidad y estandarización certificada por la comisión internacional electrotécnica (IEC).

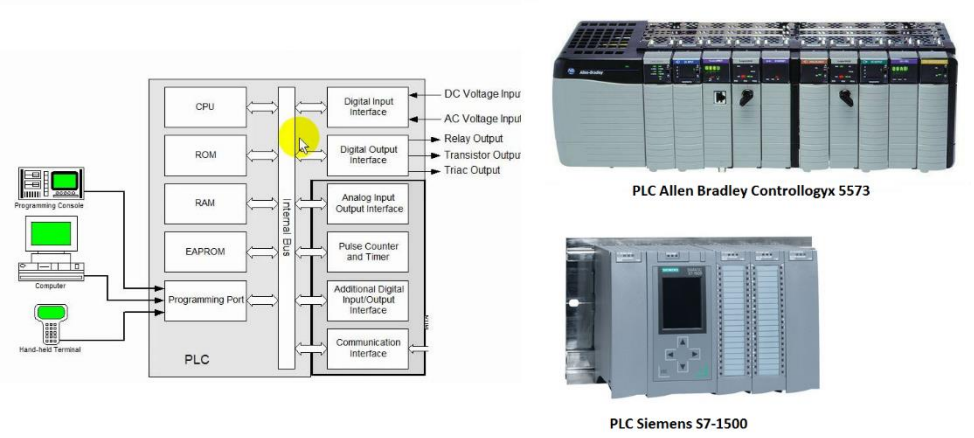


ILUSTRACIÓN 9: DIAGRAMA DE BLOQUES PLC Y MODELOS ALLEN BRADLEY Y SIEMENS

En menor medida se instalarán otras opciones como las indicadas previamente, los PC industriales o microcontroladores.

Es muy habitual encontrar trabajando de forma dual a un PLC y un PC, el primero como controlador en campo de sensores y actuadores de la máquina y el segundo como elemento integrador con otros sistemas o SCADA.

ILUSTRACIÓN 10: PANEL INDUSTRIAL CON PC INDUSTRIAL DE WECON TECHNOLOGY

Por último, los microcontroladores basados en hardware libre [HRD01]. Elementos que combinan el principio de funcionamiento del autómata programable con elementos como la interfaz Input/Output digitales y

analógicas con los microcontroladores tradicionales y la tecnología propia de lenguajes de programación como C o C++. Estos elementos trabajarán como alternativa a los anteriores o como un complemento.

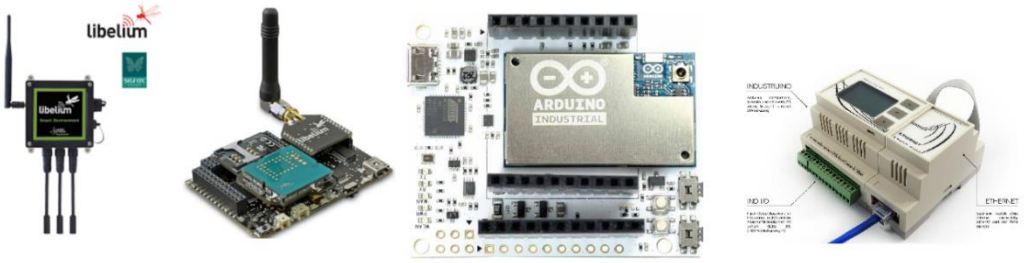


ILUSTRACIÓN 11: CONTROLADORES INDUSTRIALES OPEN HARDWARE. LIBELIUM Y ARDUINO INDUSTRIAL

2.3.3. COMPONENTES CLAVE EN LA AUTOMATIZACIÓN DE LA LÍNEA DE ENVASADO

Dada la extensión para definir y detallar los componentes clave en el área de una línea de proceso de envasado, se prepara un anexo con los elementos principales en cuanto a sensores y actuadores:

En el anexo 3 se podrá ver el estado del arte de los tipos de motores principales utilizados en las líneas de envasado.

Por otro lado, en el anexo 4, se exponen los detectores principales usados en una línea de automatización.

2.3.4. LOS CUELLOS DE BOTELLA EN LA LÍNEA DE ENVASADO

Se parte de la base de la definición de una línea de envasado como el siguiente modelo secuencial:

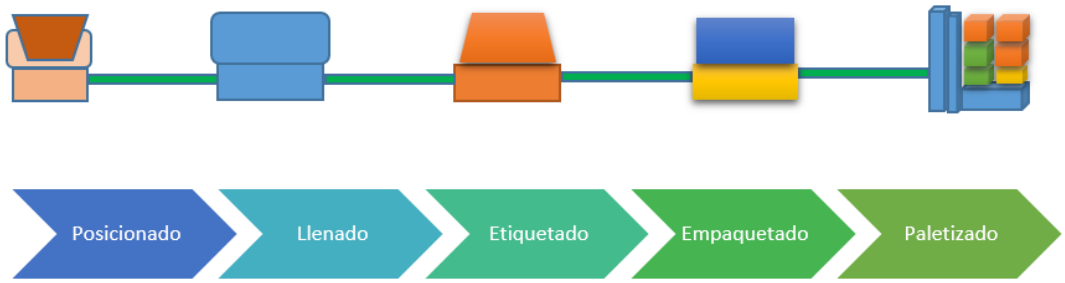


ILUSTRACIÓN 12: SECUENCIA DE LÍNEA DE ENVASADO

En el presente apartado y subapartados, se definirán las principales problemáticas que se pueden encontrar en las diferentes fases del proceso de envasado que son susceptibles de provocar cuellos de botella.

A modo resumen, se muestra el siguiente esquema con las principales problemáticas. Estos se pueden ver con mayor detalle en los siguientes subapartados:

Cuadro resumen problemáticas cuello de botella en línea de envasado		
Fase	Causa	Consecuencia
Posicionadora a envasadora	Mal posicionamiento interno de los envases	Provocará que no salgan los envases hacia la cinta de transporte que va a la máquina de envasado.
	Volumen de envase incorrecto que causa roce en su guiado a envasadora.	Caída de envases en su recorrido desde la posicionadora a estación de envasado
	Cinta transporte tiene una deformación en su guiado que hace que no avance envase vacío	

	Botella que no pertenece al modelo de formato que se está fabricando	Paralización de línea hasta que se evacuen envases.
	Problemas en subestaciones internas de estación envasado: Llenado, taponado.	Bajada de cadencia. La estación de envasado provocará el cuello de botella en la línea.
Envasadora a Etiquetadora	Cinta de transporte con deformidades o no preparada para el formato de envase.	Caída de envases en el recorrido a siguiente estación.
	Problema en cinta de transporte de Buffer: Cinta en mal estado, mal formato guías.	Caída/paralización de envases. Problema proporcional a la longitud del buffer
	Problemas intrínsecos en sellado etiquetas: Mal cambio de formato, falta de mantenimiento, exceso de cola en etiquetas.	Bajada de cadencia. La estación de etiquetado provocará el cuello de botella en la línea.
Etiquetadora a Empaquetadora	Envases con exceso de cola en etiqueta	Caída de envases o paralización de envases.
		Riesgo mal empaquetado
	Problemas intrínsecos. Mal ajuste para el empaquetado. Mantenimiento incorrecto.	Mal empaquetado en estación. Provocará paralización por avería o bajada de cadencia. Provocará cuello de botella.
Empaquetadora a paletizado	Mal posicionamiento de paquetes en transporte a paletizado por mal ajuste en cambio de formato o desajuste en salida empaquetadora	Bloqueo de transporte a estación de paletizado. Provocará cuello de botella.
	Cambio incorrecto de formato para tratar el acumulado de cajas estación de paletizado.	Bloqueo en estación.
	Problemas intrínsecos de la estación de paletizado: Mal ajuste, falta de mantenimiento.	Bajada de cadencia en estación y posible bloqueo.

TABLA 1: CUADRO RESUMEN PROBLEMÁTICAS CUELLO DE BOTELLA EN LÍNEA DE ENVASADO

2.3.4.1. FASE POSICIONADORA A ENVASADORA

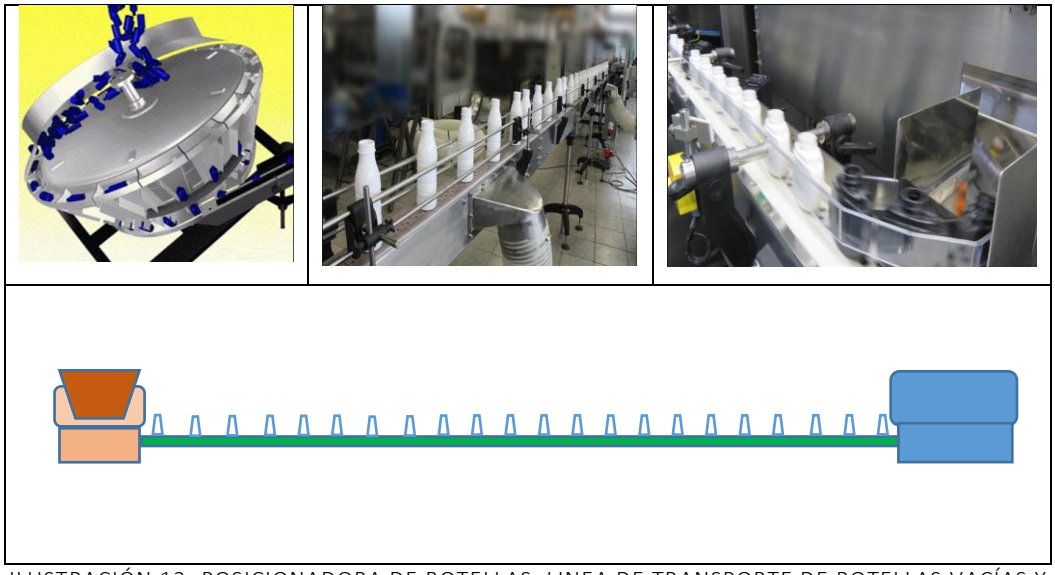


ILUSTRACIÓN 13: POSICIONADORA DE BOTELLAS, LINEA DE TRANSPORTE DE BOTELLAS VACÍAS Y ENTRADA DE BOTELLAS A ENVASADORA

En esta fase, la posicionadora depositará botellas vacías en la cinta de transporte que va a la envasadora, dado el bajo volumen y peso de la botella deberán de viajar secuencialmente hacia la llenadora sin posibilidad de generar un buffer que acumule botellas en paralelo.

Precisamente en este caso, el poco peso de la botella pueda hacer que pueda caer fácilmente si no se ha posicionado correctamente en la cinta.

¿Dónde se producen?

Se pueden producir en la máquina de posicionamiento, en la cinta de transporte y en la propia llenadora

¿Cómo se provocan?

En el caso de la **posicionadora**, dentro la misma máquina se puede provocar un mal funcionamiento debido a un desvío en sus mecanismos de posicionamiento de la botella o un desgaste de los mismos.

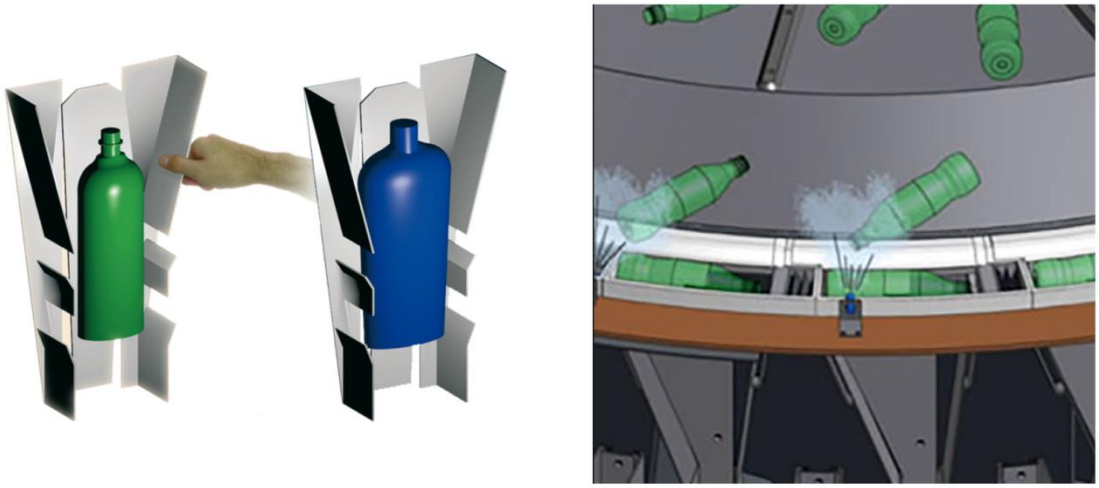


ILUSTRACIÓN 14: POSICIONAMIENTO DE BOTELLA EN OBLEAS A CINTA, ACTUADORES NEUMÁTICOS BOTELLAS EN BOMBO MÁQUINA. ® POSIMAT.

En el primer caso de la figura, si la característica de cambio de formato a botella está mal realizada y existe un mayor hueco que la botella a posicionar o al revés, el hueco es más estrecho generará un mal funcionamiento y por lo tanto un cuello de botella en la línea.

En la segunda figura, la casa de posicionadoras posimat muestra una imagen donde se pueden ver unos actuadores de aire, estos descartan las botellas mal posicionadas y hacen que la botella gire de nuevo en el tambor para poder posicionarse correctamente. Un incorrecto funcionamiento del rechazador puede provocar un bucle que no llegue a posicionar botella y por lo tanto ni que la cinta de transporte trabaje vacía de carga.

En el caso de la **cinta de transporte**, un incorrecto mantenimiento donde no exista una linealidad en el transporte puede provocar que la botella caiga y provoque un bloqueo, ante esta indisponibilidad en ocasiones puede haber un desperdiciador de botellas, quitando la botella tumbada de la cinta de transporte.

Pero es evidente que no se puede colocar un rechazador de botellas tumbadas en cada palmo de línea, este se pondrá en la posición de la cinta de transporte donde se puedan caer la botella con mayor frecuencia.

Una colocación incorrecta del detector y posterior recahazador de botellas caídas provocará una acumulación de envases incorrectos y cuello de botella.

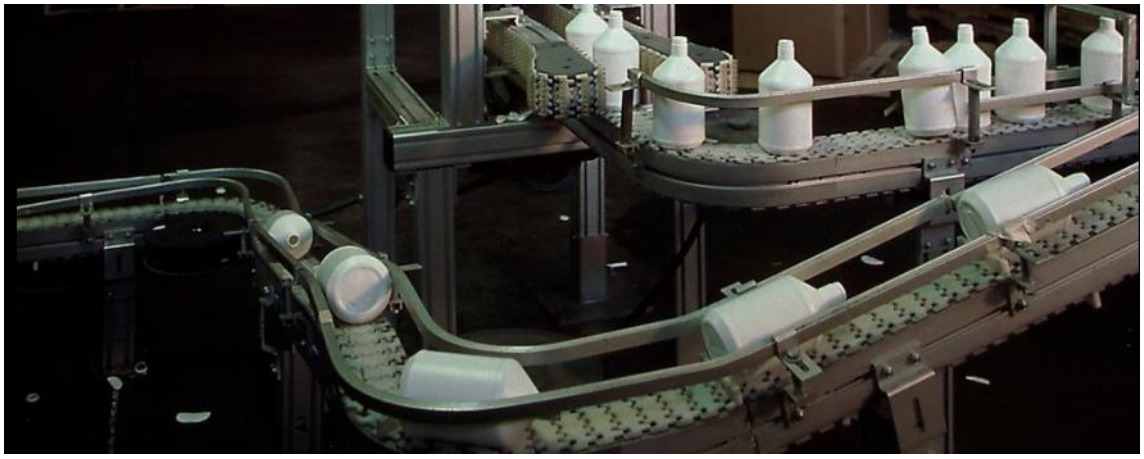


ILUSTRACIÓN 15: BOTELLAS MAL POSICIONADAS EN CINTA DE TRANSPORTE

¿Cuándo se producen cuellos de botella?

Tal y como se ha visto, alguno de los casos en los que se puede producir un cuello de botella son:

- Mal posicionamiento interno de la máquina posicionadora, provocará que no salgan las botellas hacia la cinta de transporte que va a la máquina de envasado.
- Caída de botellas en su recorrido desde la posicionadora a línea de envasado.
- Botella que no pertenece al modelo de botella del formato que se está fabricando, con lo cual provocará problemas en el envasado por sus dimensiones.

¿Por qué se producen los cuellos de botella?

- En el caso de un mal posicionado para que salga la botella puede ser provocado por una desviación en los elementos que permiten el posicionado.
- Las caídas de las botellas se pueden producir por diversos factores como un volumen incorrecto que roce por ejemplo con los laterales que guían

la botella a la envasadora o, por ejemplo, porque la cinta tiene una deformación en su base que hace que caiga el envase vacío.

Ante la persistencia de este tipo de casos existen otros medios que conduzcan la botella como el de la siguiente figura [AER01]:

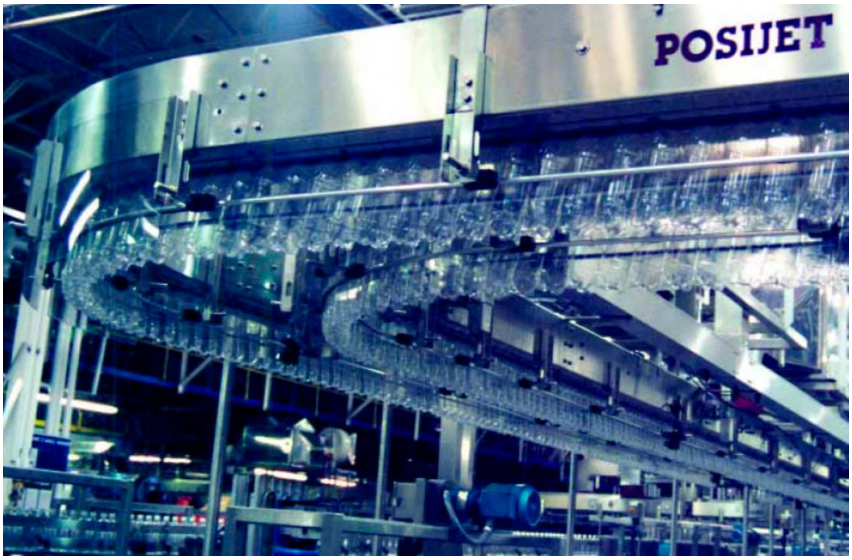


ILUSTRACIÓN 16: TRANSPORTE AEREO DE BOTELLAS ®POSIMAT

Respecto a las máquinas de llenado, tal y como se vio en el capítulo 2.3.1. De este documento estas pueden presentar una construcción lineal o rotativa.

Según su construcción y necesidad de envasado estas estaciones contendrán como mínimo una subestación de llenado y taponado del envase, en ocasiones sellado y taponado como por ejemplo en los envases de leche, presenta un aspecto de modularidad como la siguiente figura:

ILUSTRACIÓN 17: MÁQUINA DE ENVASADO BOTELLA PLÁSTICO [COM01]

La máquina queda divida en dos módulos, en primer lugar, uno de llenado y finalmente otro de taponado.

El cuello de botella se generará principalmente cuando por algún motivo tenga que trabajar más lenta por algún problema en las subestaciones internas.

La bajada de cadencia provocará un ralentizado en toda la línea de producción, quedando esta estación como cuello de botella.

2.3.4.2. FASE DE LLENADORA A ETIQUETADORA

ILUSTRACIÓN 18: FASE DE ENVASADO A ESTACIÓN DE ETIQUETADO

Una vez se ha envasado el producto, el siguiente destino será el etiquetado, para este caso, se parte de la base que la máquina de envasado contiene una subestación de llenado de producto y otra de taponado y las botellas salen por la cinta de transporte con un volumen que no tenía previamente.

ILUSTRACIÓN 19: TRANSPORTE DE BOTELLAS HACIA ETIQUETADORA

Por lo tanto, el transporte envasado tendrá mejor estabilidad en el transporte ante caídas de las botellas que provoquen atascos.

La complejidad en la máquina variará dependiendo el número de etiquetas que tenga que poner sobre el envase donde variará también la cadencia en condiciones normales.

ILUSTRACIÓN 20: MÁQUINA MULTIETIQUETADO AUTOADHESIVO. ® SIDEL

La siguiente máquina de sellado autoadhesivo contiene diversas subestaciones que dispensan la etiqueta, cuando más posiciones de etiquetado mayor probabilidad de problemas existirán respecto a máquinas que precisan solo una etiqueta como la siguiente figura [KHS01]:

ILUSTRACIÓN 21: MÁQUINA DE UNA SUBESTACIÓN DE SELLADO

En caso donde la cadencia de la máquina de etiquetado sea mayor que la estación de envasado precisará un almacenamiento previo de botellas como el que se indicó en el apartado 2.3.2, este pulmón permitirá que haya un equilibrio en la línea y no se generen acumulaciones.

¿Cuándo y por qué se producen cuellos de botella?

En el caso del envasado a la etiquetadora se podrán producir cuellos de botella por:

- Caída de botellas en el recorrido de la botella desde el envasado a línea de sellado.
- Problemas generados en buffer de entrada de la etiquetadora, donde una caída de botella puede encadenar un atasco sobre el resto o fallo de alguna de las múltiples cintas que lo componen.
- Problemas intrínsecos en el sellado de etiquetas por errores múltiples de desajustes como por ejemplo un mal cambio de formato o desviaciones provocadas por mal mantenimientos en los elementos de la máquina.

2.3.4.3. FASE DE ETIQUETADORA A EMPAQUETADORA

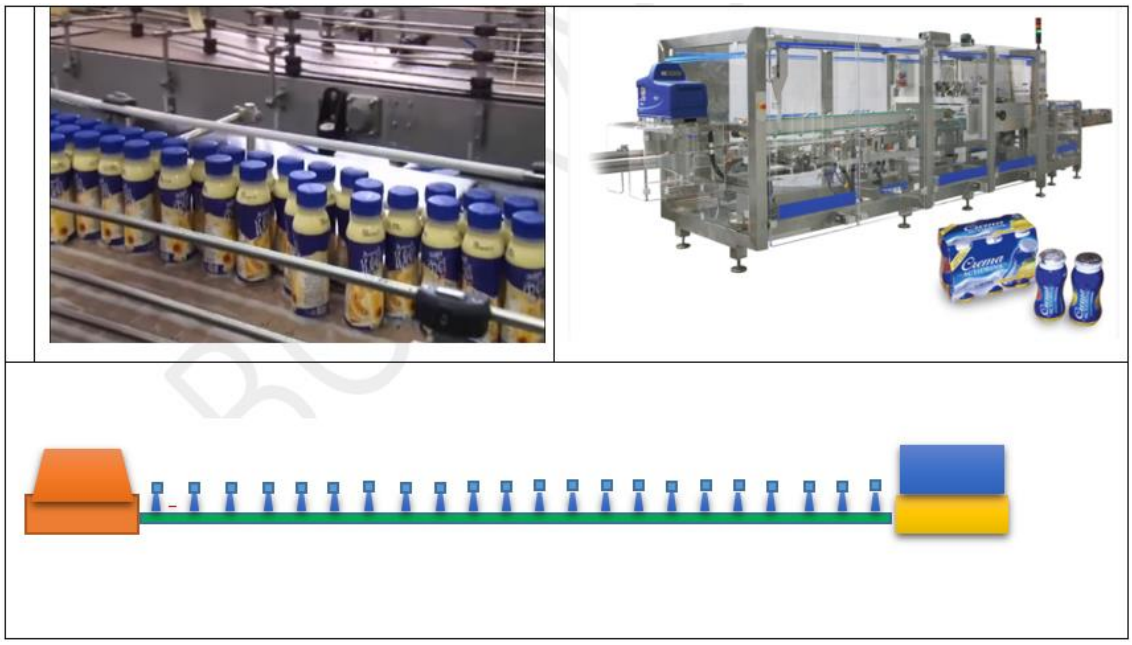


Ilustración 22: Transporte de envases etiquetados a estación de empaquetado

Una vez se han etiquetado los envases, estos serán transportados a la siguiente estación de empaquetado.

La cinta de transporte sigue la misma característica que el transporte anterior de la línea de envasado a etiquetadora, en este caso portará el envase la etiqueta.

Al igual que la fase anterior y dependiendo de la cadencia de la máquina de empaquetado, podrá existir un buffer o pulmón para mantener un nivel de envases de entrada.

Por lo tanto, el cuello de botella responderá a las mismas características que la fase anterior a diferencia de las problemáticas intrínsecas de la estación de empaquetado.

¿Cuándo y por qué se producen cuellos de botella?

En el caso de la etiquetadora al empaquetado podrán producir cuellos de botella por:

- Caída de botellas en el recorrido de la botella desde el envasado a línea de sellado.
- Problemas generados en buffer de salida de la etiquetadora, donde una caída de botella puede encadenar un atasco sobre el resto o fallo de alguna de las múltiples cintas que lo componen. Otra causa podría ser una etiqueta con restos de cola que dejara restos en la cinta de transporte o mal etiquetada que se despegara en el transcurso.
- Problemas intrínsecos en el empaquetado del producto por errores múltiples de desajustes como por ejemplo un mal cambio de formato o desviaciones provocadas por mal mantenimientos en los elementos de la máquina.

ILUSTRACIÓN 23: DETALLE DE EMPAQUETADO DE ENVASES

En la imagen se pueden apreciar dos puntos en las actividades del empaquetado del producto, la primera el posicionamiento del cartón y la segunda el producto agrupado en un pack a la salida de la máquina, en el video [CAM01] se puede ver las fases internas de la empaquetadora donde se puede apreciar cómo se agrupa el pack y posteriormente como se almacena.

2.3.4.4. FASE DE EMPAQUETADORA A PALETIZADOR

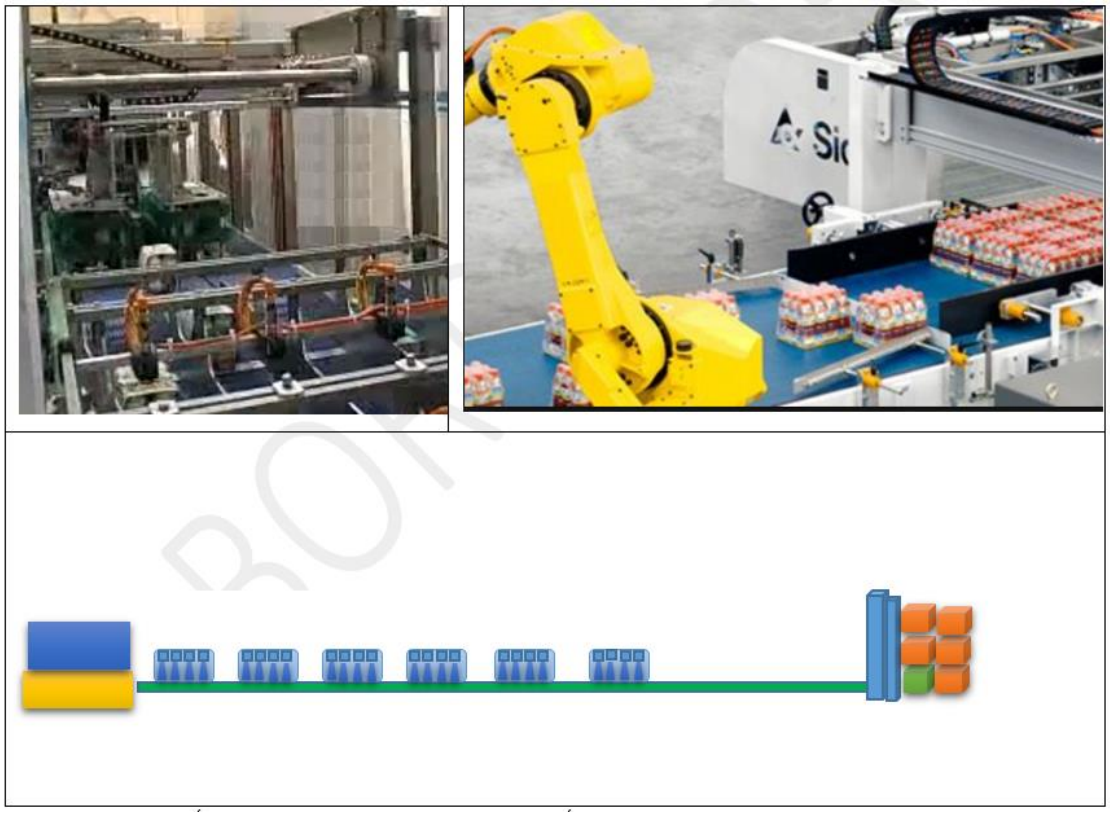


ILUSTRACIÓN 24: TRANSPORTE DESDE ESTACIÓN DE EMPAQUETADO HACIA ALMACENADO

En esta fase, transición de las estaciones de empaquetado al paletizado, el producto saldrá empaquetado según el formato de fabricación. El producto será distribuido a través de una cinta para que la estación de paletizado (ver punto 2.3.15 de este documento) finalice el proceso de envasado.

Tal y como se puede ver en la imagen de la izquierda la cinta de transporte puede distribuir los paquetes por diversas vías para prepararlos para un almacenamiento óptimo de paquetes.

A diferencia de las fases anteriores, el producto se traslada a la siguiente estación con los envases empaquetados. El control por lo tanto ya no es a nivel de envase como en las transiciones anteriores.

De cara a preparar el producto para su almacenamiento final, los envases empaquetados se pueden distribuir en diferentes vías separadas.

En la figura de la izquierda, unos sensores fotoeléctricos efectúan el contaje de los paquetes de paso y un algoritmo el que decide desviar los paquetes por las cintas para que un robot o un sistema de almacenamiento vaya apilándolos de una forma ordenada.

¿Cuándo y por qué se producen cuellos de botella?

Entre otros, los siguientes problemas son susceptibles de provocar un cuello de botella en la presente fase:

- Bloqueo de paquetes provocados por un mal posicionamiento en las cintas de conducción hacia el almacenador.
- Un mal ajuste de formato en la salida de la estación de empaquetado.
- Cambio incorrecto de formato para tratar el producto en las máquinas asociadas a esta fase.
- Problemas intrínsecos de la estación de paletizado que generen ralentización al resto de estaciones.

En general, se ha visto que muchos problemas pueden venir relacionados con un mal ajuste en las estaciones de una línea de envasado en el momento en el que se hace un cambio de formato de producto.

2.4. SOLUCIÓN PROPUESTA PARA MEJORAR LA SITUACIÓN

2.4.1. INTRODUCCIÓN

Tal y como se ha visto a lo largo del contexto de las líneas de envasado respecto a los cuellos de botella, existen muchas situaciones donde se pueden generar cuellos de botella y estas dependen de diversos factores como por ejemplo un mal cambio de formato en las estaciones de trabajo o un mantenimiento incorrecto en sus componentes.

De la misma forma, también se ha visto que existen muchos tipos de maquinaria para cubrir las diversas estaciones de trabajo, por esta razón para poder cubrir una iniciativa de mejora, se escogen una serie de variables comunes que permitan detectar un cuello de botella a través de las tecnologías orientadas al internet de las cosas y las aplicaciones basadas en el Web actuales.

Tal y como se vio en el apartado 2.3.3 de este documento, las estaciones de trabajo están gobernadas por sistemas automatizados orientados a su regulación, control y su supervisión. Los fabricantes ofrecen una automatización en ocasiones a medida, pero acotada a la estación de trabajo.

Para mejorar la sincronización entre estaciones muchas veces es necesario un esfuerzo de cada fabricante para dejar un subsistema/módulo del programa que permita enviar información de su cadencia y otras variables que permitan autorregular la cadencia de la estación anterior y posterior, aunque este aspecto dependerá si el fabricante lo permite. Obviamente, si todas las estaciones de trabajo son de un mismo fabricante este servicio estará dispuesto.

La solución propuesta permite cubrir la detección inteligente de datos clave que puedan provocar posibles cuellos de botella, para ello el sistema se sustenta sobre una serie de variables iniciales:

- Cadencia de máquina. Estipula el tiempo medio de salida de los productos de cada estación. [UPM01, Cadencia de fabricación]. Cada fabricante indica la cadencia Máxima de máquina trabajando en condiciones normales sin que sufran los componentes.
  La medición se efectuará a partir del número de botellas fabricadas a la hora. (Sensor en la salida de máquina)
- Número de recipientes de entrada en un determinado intervalo de tiempo. Este puede resolver información estadística como medias, desviaciones, tiempos máximos y mínimos entre detección etc.
- Datos del controlador principal de la estación y transporte como rechazos de envases, velocidad media, número de paradas efectuadas, etc.
- Trazabilidad de envases a partir de un sensor inteligente RFID.
- Adquisición de datos clave de los sistemas de gestión de mantenimiento de planta

La idea es obtener un sistema global a través de una pila de componentes que permita obtener una información de cuellos de botella y que permita tener una serie de datos para tomar decisiones a nivel de Operaciones en una línea de fabricación.

En definitiva, la propuesta es un posible análisis con una serie de variables que permitan conocer datos para poder reconocer los posibles cuellos de botella en un tiempo determinado y se puedan tomar decisiones al respecto.

2.4.2. QUÉ Y CÓMO SE PRETENDE MEJORAR LA SITUACIÓN

Según se pudo ver en el capítulo 2.2 donde se investiga el estado del arte en los cuellos de botella, los autores mostraban sus pautas para poder gestionar estas problemáticas, entre ellas, las dos siguientes:

- [ATO01] La importancia en medir las variables de rendimiento clave en un cuello de botella.

- [PLA01] Mencionaba, que era necesario gestionar de forma continua los cuellos de botella, monitorizarlos.

El proyecto seguirá estos dos principios para tratar de mejorar la situación en cuanto a los cuellos de botella en esta situación investigada en las líneas de envasado de productos de plástico.

El objetivo primordial y por lo tanto la respuesta al "que", es obviamente mejorar el cuello botella en la línea de producción, a través de las pautas indicadas en el párrafo anterior a partir del esquema general basado en la medición, gestión de las variables y su monitorización y para toma de decisiones.

ILUSTRACIÓN 25: FLUJO DE INFORMACIÓN

Esto se pretende mejorar a través de la utilización de las tecnologías que ofrece el paradigma IoT y el paradigma Web, ayudándose de disciplinas como la generación automática de código.

Para ello se creará un sistema basado en componentes hardware y software que permita poder llevar a cabo el objetivo de mejora de la problemática. Una iniciativa con la idea que a partir de sus datos tratados sirvan para que puedan tomar decisiones desde los departamentos de Operaciones de la línea productiva pueda encontrar soluciones.

El siguiente cuadro muestra la iniciativa en forma de bloques desde un punto de vista de negocio:

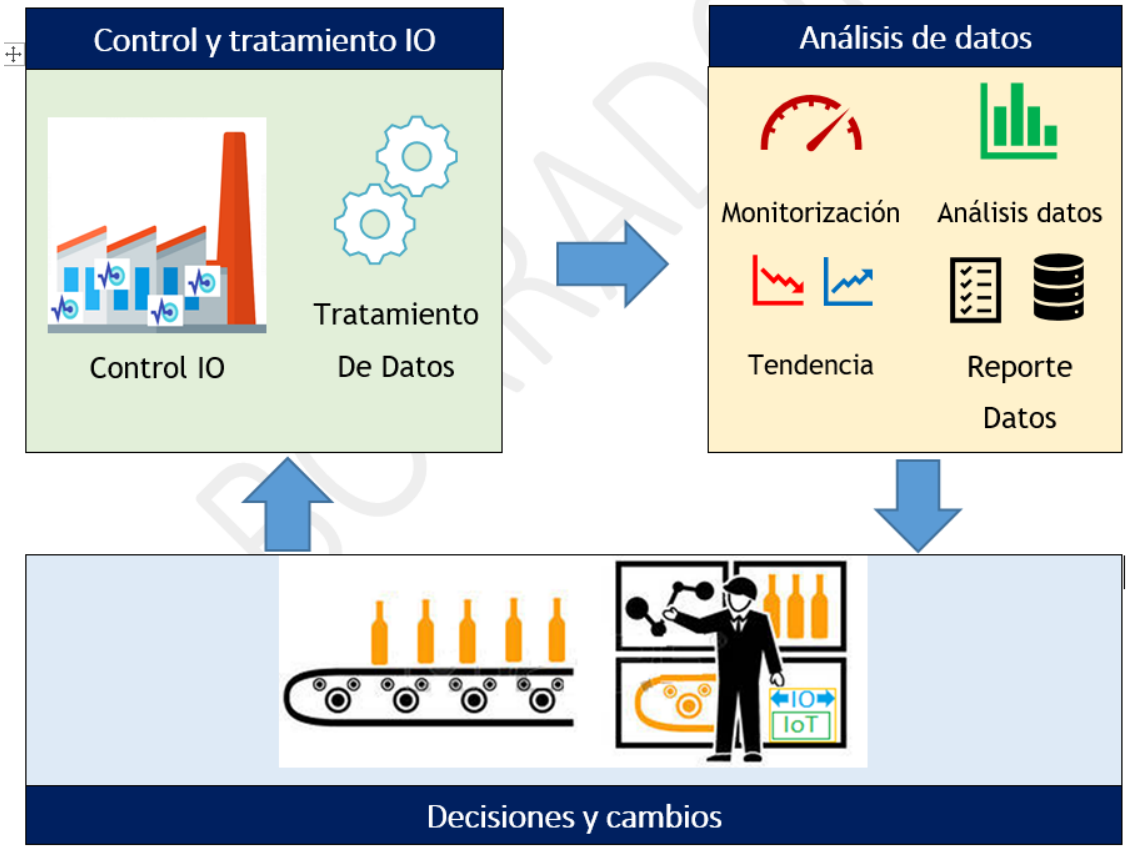


Ilustración 26: Cuadro IoT para la iniciativa de análisis y decisión sobre los datos para mejorar la situación.

El bloque de control y tratamiento IO, está formado por los elementos de campo que generan los datos de entrada/salida y el tratamiento de estos para que pasen al bloque de análisis de datos donde se generará una serie de información que servirá como input para el bloque de decisiones y cambios donde se concluirá sobre los datos obtenidos y analizados y se decidirá actuaciones de nuevo en el bloque de control y tratamiento IO (Sensores y Actuadores).

Para concretar, el bloque de análisis de datos se compone de cuatro elementos de información que deberán permitir tomar decisiones:

- Monitorización: Permite verificar las variables clave medidas:
    - Cadencia de estación de trabajo (envases hora), detección de envases por unidad de tiempo en zonas importantes de máquina, velocidad de

- Detección de envases en distintos puntos importantes en la estación de trabajo.
- Velocidad de estación y cintas de transporte de entrada y salida.
- Valores por cada estación de trabajo.

- Estadística y tendencias: Valores medidos en intervalos de tiempo semanales, mensuales, anuales que permitan ver valores Máximos, mínimos y medias de las variables monitorizadas
- Datos: Histórico de datos guardados que permitan comparar con los actuales.

2.4.3. SITEMA IOT

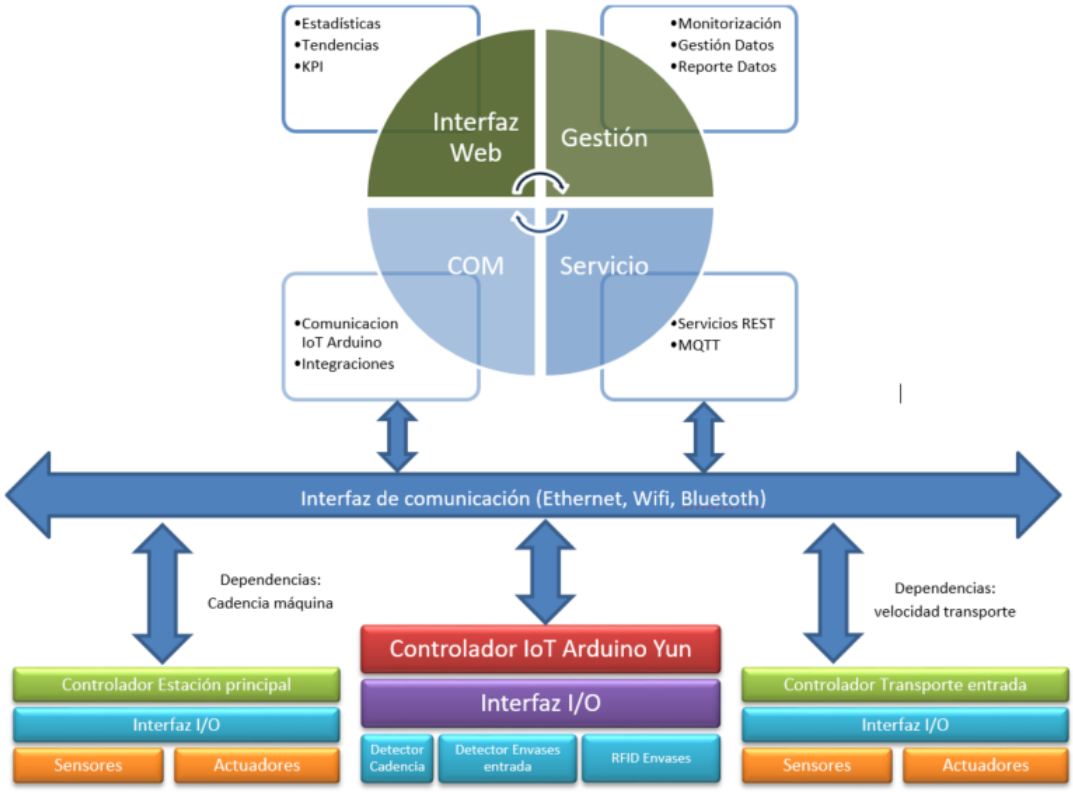


ILUSTRACIÓN 27: ARQUITECTURA GENERAL DEL SISTEMA IOT

Según se puede ver en la imagen, el sistema IoT se compone de una serie de elementos cohesionados entre sí en tres bloques principales:

- Controlador IoT donde se adquieren los tres datos principales: Detección de cadencia, detección de envases de entrada y lector RFID de envases.
- Dependencias externas: Cadencia de máquina provista por el controlador principal de la estación de trabajo y velocidad de transporte provista por su controlador principal.
- Interfaz de comunicación: El sistema de comunicación que enviará los datos principalmente a través de tres vías (Ethernet, WiFi y Bluetooth)
- Interfaz de usuario: Sistema Web que gestionará los datos adquiridos para diferentes finalidades: Estadísticas, tendencias, monitorización y las funcionalidades necesarias para que se puedan tomar decisiones para acotar la problemática analizada.

2.4.3.1. RESUMEN DE VARIABLES QUE TRATARÁ EL SISTEMA

- Cadencia de máquina: Esta información se adquiere desde dos fuentes:
  - Detector en salida de estación, mide las botellas a la hora.
  - Dato adquirido desde el sistema de control principal de la estación de trabajo.
- Detección de envases de entrada:
  - Detector en la entrada de estación. Puede verificar posibles caídas de botella, retraso en la estación previa. Toma como referencia la detección media entre botellas.
- Velocidad de estación de trabajo:
  - Velocidad de máquina adquirida desde el control principal de máquina.
- Velocidad de cinta de entrada:
  - Información adquirida desde el sistema de control de la cinta de transporte.

- Control de botellas de entrada y salida: Antena RFID en entrada y salida de estación. El envase debe de estar preparado con una etiqueta RFID donde se adquiera su ID.
- Botellas rechazadas: Botellas que no han sido aceptadas por defectos detectados.

De forma resumida, el sistema IOT para dar solución a la problemática estará compuesto por los siguientes componentes:

Sistema de control IoT de hardware libre Arduino Yun

Con una interfaz de entradas y salidas A/D, tiene la responsabilidad de adquirir la información de los sensores de campo.

Dato	Interfaz IO	Tipo	Descripción
Cadencia de máquina	Input digital	Fotosensor	Detector en salida de estación, mide las botellas a la hora.
Detección envases entrada estación	Input digital	Fotosensor	Detector en la entrada de estación. Puede verificar posibles caídas de botella, retraso en la estación previa. Toma como referencia la detección media entre envases.
Control/traza de botellas	Input digital	Input RFID	Antena RFID en entrada y salida de estación. El envase debe de estar preparado con una etiqueta RFID donde se adquiera su ID.

TABLA 2: DATOS DEL SISTEMA DE CONTROL IOT DE HARDWARE LIBRE ARDUINO YUN

Dependencias / Integración con otros sistemas

Datos que susceptibles de adquirir de los sistemas que controlan la estación de trabajo y la cinta de entrada.

Dato	Tipo integración	Tipo comunicación	Descripción	Fuente
Cadencia de máquina	Integración entrada		Dato adquirido desde el sistema de control principal de la estación de trabajo.	Estación de trabajo
Velocidad estación trabajo	Integración entrada		Variación de Velocidad de estación	Estación de trabajo

Velocidad cinta entrada	Integración entrada	Variación de Velocidad del de cinta entrada	Cinta entrada
Número de botellas no aceptadas	Integración entrada	Dato de número de botellas no aceptadas entrada estación.	Estación de trabajo
Número de paquetes no aceptados	Integración entrada	Dato de número de paquetes no aceptados entrada estación.	Estación de trabajo
Listados de errores	Integración entrada	Datos relacionados con los errores producidos en la estación y cinta de entrada	Estación de trabajo / Cinta de entrada

TABLA 3: INTEGRACIÓN DE DATOS CON OTROS SISTEMAS EXTERNOS

Sistema Web

En líneas generales, a través de este sistema basado en tecnología Web, se adquiere la información del hardware IoT y las dependencias externas al sistema para hacer el tratamiento de datos y la monitorización de los mismos tal y como se mostró en el capítulo 2.4.

ILUSTRACIÓN 28: COMPONENTES SISTEMA WEB

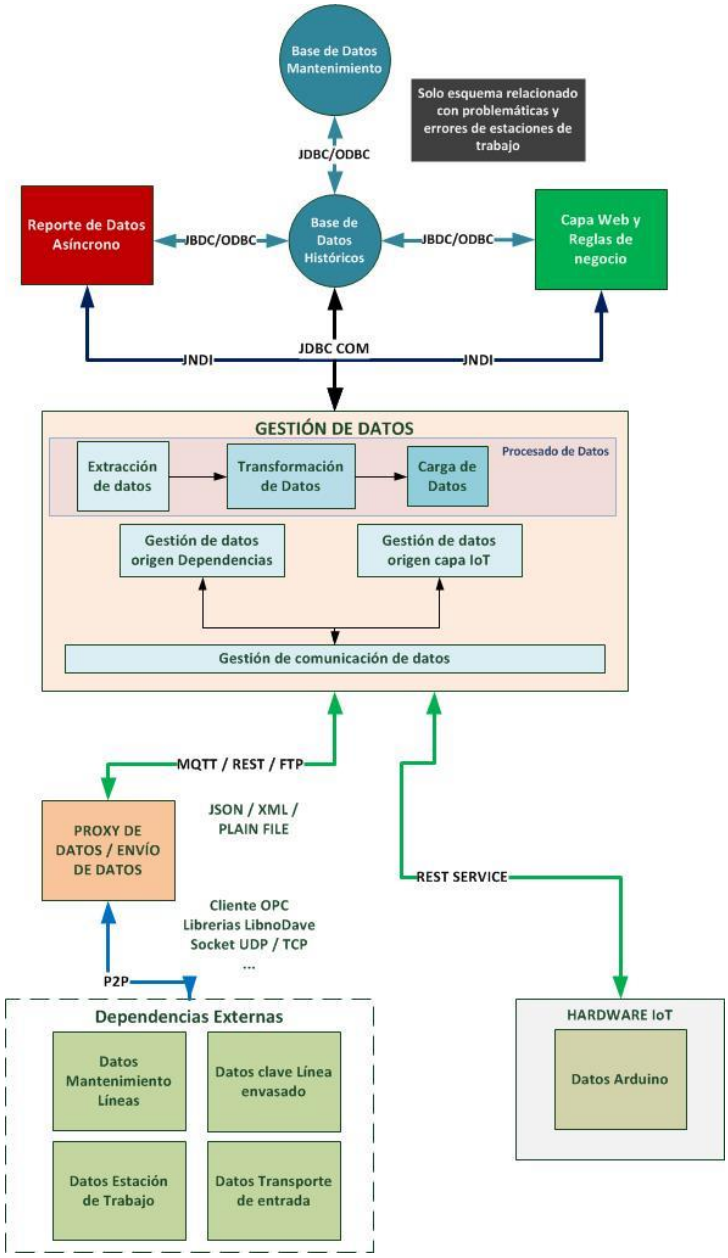


ILUSTRACIÓN 29: COMUNICACIÓN ENTRE BLOQUES DEL SISTEMA IOT

La ilustración muestra una visión global de cómo se intercomunicarán los diversos bloques o subsistemas. En la especificación técnica se verá esta interrelación con mayor detalle tanto a nivel de infraestructuras como a nivel de componentes tecnológicos de software (apartados 3.4.1 y 3.4.2 respectivamente).

2.4.4. BLOQUE RESUMEN DE COMPONENTES FUNCIONALES

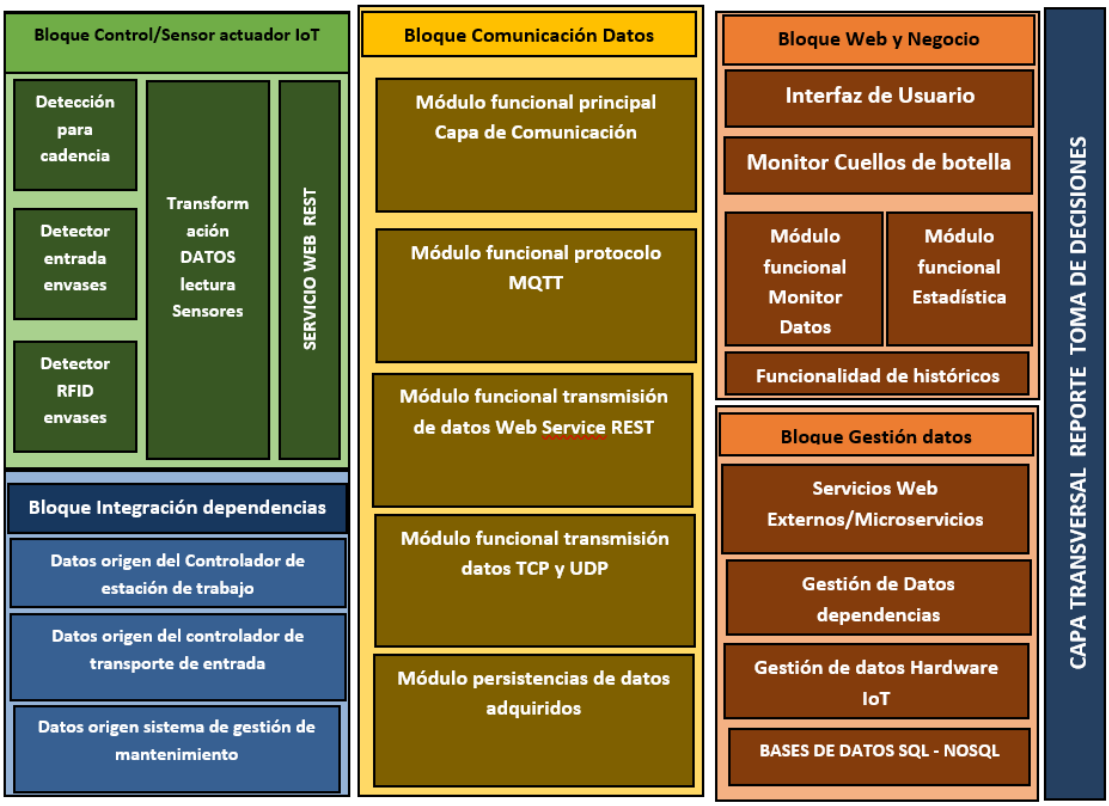


ILUSTRACIÓN 30: PILA DE COMPONENTES FUNCIONALES

La figura muestra la pila de componentes funcionales del sistema, esta se basada en los siguientes cinco bloques:

Bloque de Control/sensor actuador IoT

El sistema comprende una capa a nivel de hardware IoT donde estarán conectados tres elementos de entrada:

- Detector de cadencia. Se instala un detector de envases en la salida de máquina para cuantificar las botellas horas que salen. Los datos obtenidos se contrastarán con el dato de cadencia obtenido desde el sistema de control principal de la estación de trabajo.

- Detector de entrada de máquina. Se instala un detector de envases en entrada de máquina con el objetivo de medir el tiempo entre cada envase y poder compararlo con el tiempo medio óptimo.
- Detector RFID. Si la línea de envasado permite botellas con etiqueta RFID se instalará un sensor para tener una traza de la botella sobre su identificador.

En esta capa los datos serán transformados en datos transportables a partir de Servicios Web bajo API REST.

Bloque de Integración de dependencias

Se llaman dependencias a los sistemas que no forman parte del hardware IoT donde se precisa información de la estación de envasado. La información se adquiere de los sistemas de control de estas dependencias.

Para el presente sistema y por estación, existirán dos dependencias.

- Datos relevantes de la estación de trabajo:
    - Valor de cadencia de máquina.
    - Valor de botellas expulsadas a la hora por no cumplir las condiciones de entrada (Caso de estaciones cuya entrada sea el envase).
    - Paquetes expulsados (Caso de estacione cuya entrada se un paquete con un conjunto de envases empaquetados).
    - Velocidad estación de trabajo.
    - Datos relevantes de mantenimiento para la detección de cuellos de botella.

Se reciben los datos a través de un bloque de comunicación enlazada con los controladores principales de la estación de trabajo y cinta de entrada.

Bloque de Comunicación de datos

En este bloque se extraerán y transformarán los datos provenientes de las dependencias externas y del hardware IoT para que puedan ser según el caso

utilizados para los componentes del bloque Web y Negocio y el bloque de gestión de datos.

Según se puede ver en la ilustración, se prevén diversos módulos especializados según sea el origen de datos. Un módulo principal que abstraerá la lógica principal de comunicación y otros módulos para tratar de forma específica diversos protocolos de comunicación (REST, MQTT, TCP WEBSOCKET).

Bloque Web y Negocio

En este bloque se instalarán las funcionalidades para cumplir el objetivo de monitorizar los datos, estadísticas y datos relevantes que permitan poder tomar decisiones desde una visión para acotar la problemática sobre los cuellos de botella.

Bloque Gestión de Datos

El bloque de gestión de datos es fundamental para el envío de estos a las Bases de datos tanto históricos como datos instantáneos y con ello poder ser servidos a la capa Web. Para la generación de reportes se utilizarán las herramientas vistas en la generación automática de código para transformar estos datos en una salida que pueda consumir La capa Web y de Reportes ya sea de forma síncrona como asíncrona para que el usuario pueda tomar decisiones ante la problemática de los cuellos de botella.

También existirá una capa de microservicios para enviar y una capa cliente para recibir información de otras estaciones de trabajo.

Bloque transversal de reporte de Datos

Este bloque consumirá los datos transformados de la capa de gestión de datos y generará de forma automática reportes con indicadores clave (KPI) del sistema para acotar los problemas de cuellos de botella.

2.5. PROYECCIÓN DE LA SOLUCIÓN PROPUESTA

De forma general, la solución y metodología propuesta en el presente proyecto, servirá para que los usuarios puedan tomar decisiones sobre los datos adquiridos ante la presencia de cuellos de botella en sus procesos productivos.

El presente trabajo, se proyecta como una iniciativa a aplicar sobre los procesos productivos de líneas de envasado sobre un sistema basado en tecnología IoT y Web. Independiente en cuanto al control de la línea de envasado (ya sea de forma global o modular por cada estación) pero que precisará información de estas para adquirir información sobre las variables mencionadas en el punto anterior.

El objetivo principal, es reducir los cuellos de botellas ante la información que proporcione este sistema. La forma que se produce esta, es por un lado Online -no instantánea-, es decir con cierto retardo que provoque las limitaciones tecnológicas y, por otro lado, en forma de un procesado con información guardada e histórica para poder generar reportes e indicadores que permitan tomar decisiones.

A partir de aquí, la puesta en servicio de este sistema y un periodo de pruebas determinado podrá corroborar la hipótesis inicialmente propuesta para poder mejorar la problemática planteada.

2.6. RELACIÓN CON LOS TEMAS ESTUDIADOS EN COMPUTACIÓN UBICUA

La disciplina de computación ubicua ha permitido adquirir conocimientos en cuanto a los sistemas IoT actuales.

Es por esto, que a lo largo del estudio de la disciplina y sus módulos de trabajo se desglosó el funcionamiento de un sistema IoT -como una de las contribuciones en la computación ubicua- desde una visión global hasta el

conocimiento concreto en cuanto a hardware, sensores, sistemas de servidores, comunicaciones y en definitiva un ecosistema de computación ubicua.

Para el presente trabajo se aprovechan todos estos conocimientos a partir de la aplicación de un sistema IoT que permite una iniciativa de solución a las problemáticas en los cuellos de botella.

El sistema de esta solución cohesiona un conjunto de elementos relacionados con esta disciplina, particularmente estos elementos tienen relación con los tres módulos que se estudiaron en Computación Ubicua:

- Fundamentos, donde se ofreció una visión para poder idear un sistema global como el que se pretende diseñar en este trabajo.
- Tecnologías, a partir de las cuales, se permitieron ver la composición de sistemas inteligentes y una visión de las tecnologías aplicadas en los sensores. Sobre este módulo se basa la utilización del hardware open source, sensores para adquirir información de las estaciones de trabajo, la comunicación y el servidor de aplicaciones que orquesta la información.
- Contribuciones, en definitiva, el resultado de todo lo anterior permite construir un sistema basado en internet sobre los elementos y admite una vía para conseguir un sistema como el que se pretende mostrar en este trabajo.

2.7. RELACIÓN CON LOS TEMAS ESTUDIADOS EN GENERACIÓN AUTOMÁTICA DE CÓDIGO

La relación con la disciplina de Generación Automática de Código viene dada a partir del procesado de datos que se ha de realizar con los datos adquiridos desde las capas de control de Sensores y Actuadores del sistema Hardware IoT y de la integración con las dependencias de otros sistemas.

La disciplina permitió adquirir conocimientos en cuanto a cómo se pueden procesar de forma automática tanto código como datos para transformarlos en una solución, para este trabajo se utilizan en concreto los módulos vistos:

- Generadores simples y para gestores de datos. Estos apartados permitieron ver como se modula un proceso a través de fases simples para generar una salida a una problemática propuesta.

  En el contexto del sistema solución de esta investigación, los datos podrán venir en formatos como JSON, XML, fichero plano y el sistema generará un output para sus clientes tanto el servidor de aplicaciones como para procesar los datos de forma asíncrona.

- Generación automática orientada a paradigmas. En este sentido, el procesado de datos se puede orientar a una línea de productos determinada que pueda reutilizarse en otros contextos, se utilizará un proceso de extracción, transformación y carga de datos donde se extraerán los datos para proporcionar un output con el objetivo de alimentar las capas Web donde se mostrará la información Online y a la capa transversal de reportes. Por otro lado, la utilización del framework SPRING [SPR02] permite el uso del paradigma de la programación orientada a aspectos visto en este mismo módulo de GAC como opción de investigación, a nivel de este trabajo se utilizará como un método para interceptar datos como en el siguiente gráfico:

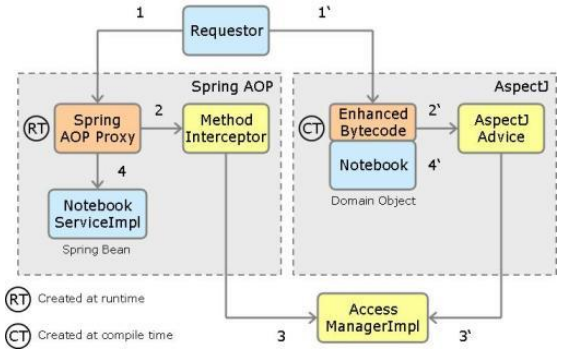


ILUSTRACIÓN 31: DETALLE IMPLEMENTACIÓN PARADIGMA AOP

3. ESPECIFICACIÓN Y DISEÑO DE LA SOLUCIÓN IOT DE MEJORA

3.1. ESPECIFICACIÓN FUNCIONAL

En el presente capítulo se desglosarán los bloques funcionales que se vieron en el apartado 2.4.4. En este apartado se resumía de forma breve la función de los siguientes bloques funcionales:

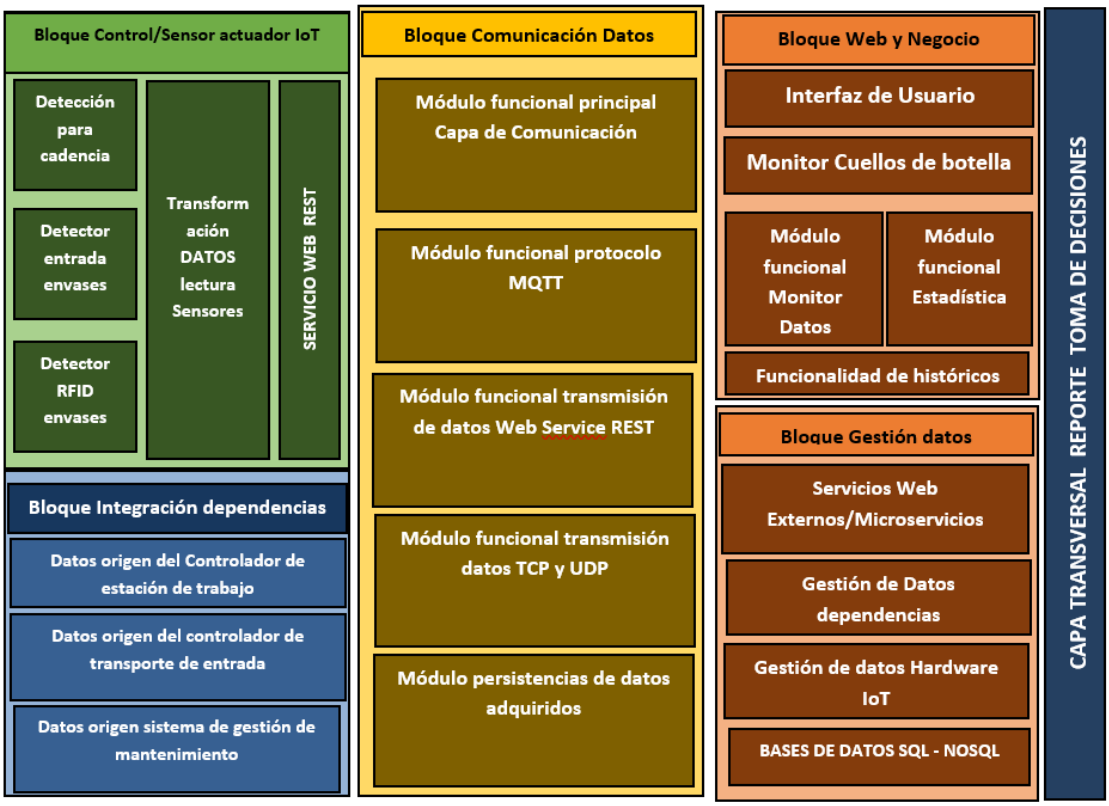


ILUSTRACIÓN 32: BLOQUES FUNCIONALES

A partir de aquí, cada uno de estos bloques encapsulará diversas funcionalidades con una serie de actividades cuyo objetivo es dar forma al sistema ideado.

En el siguiente apartado se describirán las funcionalidades que cubre cada bloque y a continuación se propondrá una ficha a través de un identificador de funcionalidad con su detalle por cada una de ellas.

3.1.1. COMPONENTES FUNCIONALES

Bloque de control Sensor Actuador IoT:

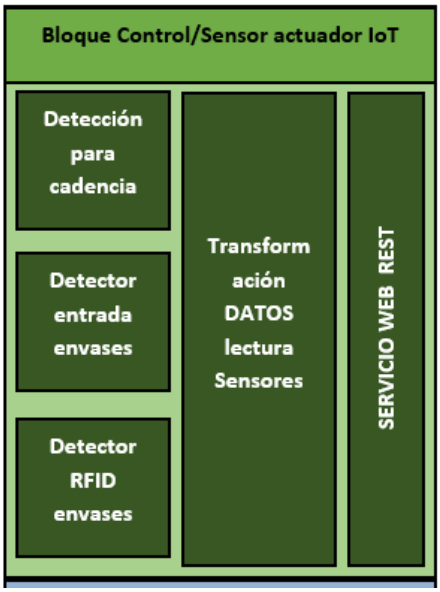


ILUSTRACIÓN 33: BLOQUE CONTROL SENSOR ACTUADOR IOT

- FU-CSA001: Funcionalidad de adquisición y transformación de datos del sensor a través de las interfaces de entradas y salidas digitales.
- FU-CSA002: Funcionalidad de gestión de datos adquiridos de los sensores de la interfaz de entradas del controlador IoT, las señales medidas se convierten en tipos de datos tratables a través de un sistema de información.

- FU-CSA003: Funcionalidad de transformación del dato en formatos de envío a través de servicios REST.

Bloque de integración de datos de dependencias:

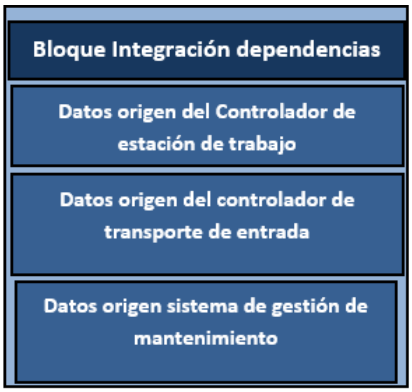


ILUSTRACIÓN 34: BLOQUE DE INTEGRACIÓN DE DEPENDENCIAS

- FU-IDD001: Funcionalidad proxy para adquisición de datos de dependencias externas.

Bloque de comunicación de datos y servicios:

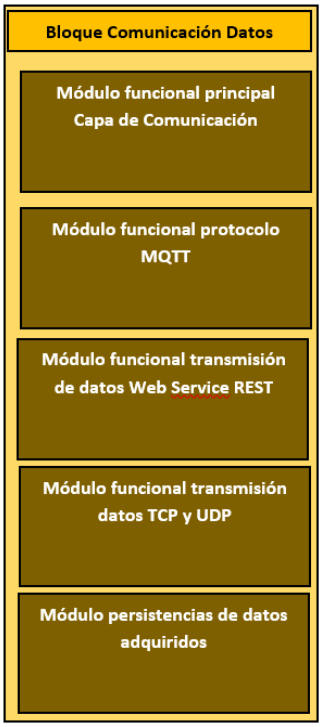


ILUSTRACIÓN 35: BLOQUE DE COMUNICACIONES

FU-COMO001: Funcionalidad principal para la gestión de comunicación.

- Extracción de datos MQTT.
- Extracción de datos Servicios REST.
- Extracción de datos transmisión Ethernet.

Funcionalidades Gestión de datos:

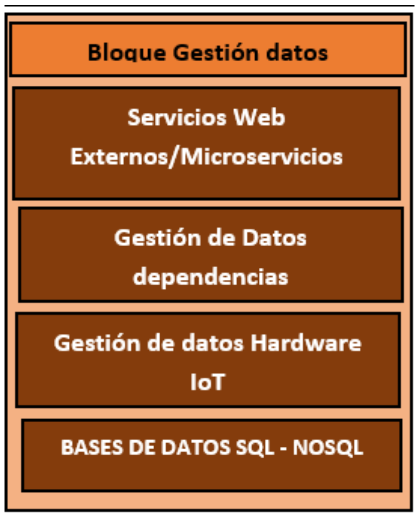


ILUSTRACIÓN 36: FUNCIONALIDADES DE GESTIÓN DE DATOS

- FU-GDA001: Gestión de datos de dependencias externas y datos hardware IoT.
- FU-GDA002: Funcionalidad de publicación de datos clave a través de microservicios.
- FU-GDA003: Funcionalidad de tratamiento y carga de datos para reporte.

Bloque funcional Web y Negocio:

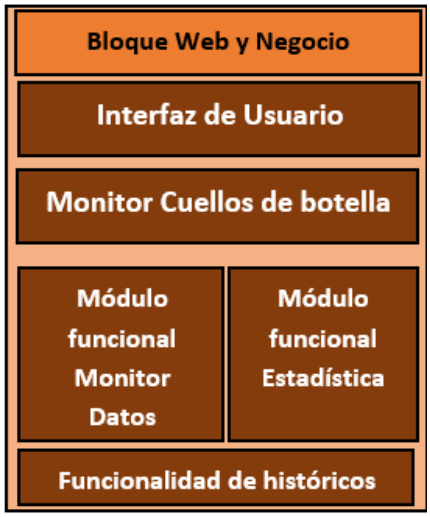


ILUSTRACIÓN 37: BLOQUE WEB Y NEGOCIO

- FU-WEB001 - Funcionalidad de Monitorización de datos clave/cuadro de mando cuello de botella estación de trabajo.
- FU-WEB002: Funcionalidad gestión de reportes. (Pantalla gestión de reportes)
- FU-WEB003: Gestión de datos históricos. (pantalla con opciones para seleccionar gráficos de históricos)

Capa transversal de reporte:

FU-RDA001: Funcionalidad para la generación automática de reporte de datos a usuarios clave del sistema

3.1.1.1. FUNCIONALIDADES CONTROL SENSOR ACTUADOR IOT

FU-CSA001: Funcionalidad de adquisición de datos del sensor a través de las interfaces de entradas y salidas digitales.

CU / Funcionalidad	Adquirir datos hardware IoT	FU-CSA001
Actores	Envases, Controlador IoT	
Tipo	Base	
Precondición	• El sensor fotoeléctrico de entrada y salida de estación y antena RFID conectada y probada eléctricamente sobre el dispositivo de control. • La línea de envasado está en funcionamiento en producción con el envío de envases desde inicio de línea a final. • Los sistemas de etiquetado y empaquetado están en condiciones de producción.	
Postcondición	El software del controlador IoT tiene lectura de los tres sensores para poder gestionar el programa de tratamiento.	

Propósito
Adquirir datos de los sensores de entrada del controlador IoT

Resumen
La funcionalidad recibe la señal de los tres dispositivos sensor indicados en la precondición. Los envases llegan por la cinta de transporte a la entrada de la estación de trabajo, a su vez estos envases hacen el ciclo completo en la estación de trabajo hasta que salen por la cinta de transporte hacia la siguiente estación de trabajo.
Flujo Base
Recibe señal de la interfaz de entradas del controlador IoT: • Los sensores fotoeléctricos instalados en la entrada y la salida de la estación envían una señal digital cada vez que detecta la presencia de un envase. • Lector RFID situado en la entrada de estación envía una señal analógica con la identificación del envase cada vez que detecta su presencia.

FU-CSA002: Funcionalidad de gestión de datos adquiridos de los sensores de la interfaz de entradas del controlador IoT, las señales medidas se convierten en tipos de datos tratables a través de un sistema de información.

CU / Funcionalidad	Gestión de datos adquiridos de los sensores de la interfaz de entradas del controlador IoT.	FU-CSA002
Actores	Sistema IoT	
Tipo	Base	
Precondición	El Software del controlador IoT tiene lectura de los datos digitales y analógicos.	
Postcondición	Se obtienen registros según las reglas mostradas en el flujo base para ser enviados vía Servicios Web.	

Propósito
Tratar y tramitar los datos adquiridos en FU-CSA001 a través de una serie de reglas de negocio y prepararlos para el envío de información para la capa de gestión de datos.
Resumen
El sistema hardware IoT a través de su controlador software trata la información adquirida de los sensores para preparar su envío a la capa superior de gestión de datos.
Flujo Base
• Tratamiento del sensor de cadencia (información del sensor de salida de estación)

- Registra y acumula la detección de envases.
- Controla el número de envase al minuto.
- Realiza conversión a envases hora.
- Prepara un registro de la cadencia registrada por minuto y se añade a un servicio Web para ser enviado.

- Tratamiento del sensor de entrada de estación RFID
  - Registra el identificador leído desde la antena RFID ante la detección de un envase nuevo.
  - Guarda el identificador en una tabla Hash con el momento de detección.
  - Prepara un vector con los registros adquiridos en un minuto y se añade a un servicio Web para ser enviado.
- Tratamiento del sensor de entrada de estación
  - Controla la detección de envases nuevos.
  - Registra las repeticiones de detección entre botellas mayor a 10 segundos.
  - Registra tiempo medio entre botellas por minuto.
  - Recoge los dos registros anteriores y se preparan para enviarlos por servicio Web a la capa superior.

FU-CSA003: Funcionalidad de transformación del dato en formatos de envío a través de servicios REST.

CU / Funcionalidad	Envío de datos tratados de los sensores de la interfaz de entradas del controlador IoT	FU-CSA003
Actores	Sistema IoT	
Tipo	Base	
Precondición	Registros preparados obtenidos en la funcionalidad FU-CSA002	
Postcondición	Se han enviado y recibido en el módulo de gestión de datos los registros enviados vía servicio Web	

Propósito
Enviar la información tratada en la funcionalidad FU-CSA002 a la capa de Gestión de datos
Resumen
El subsistema Web del controlador IoT envía un Servicio Web con los datos que se han adquirido en un minuto según el flujo base de la funcionalidad FU-CSA002.
Flujo Base
• Preparar estructura de datos en un formato serializable (XML, JSON) con los registros adquiridos por minuto.

• Preparar trigger para publicar y enviar los datos a través del servicio Web. • Publicar Servicio Web.

3.1.1.2. FUNCIONALIDADES DE INTEGRACIÓN DE DATOS DE DEPENDENCIAS

FU-IDD001: Funcionalidad proxy para adquisición de datos de dependencias externas.

CU / Funcionalidad	Gestión de datos adquiridos de las dependencias estación de trabajo	FU-IDD001
Actores	Sistemas externos	
Tipo	Base	
Precondición	El software controlador principal de estaciones de trabajo y de cinta de entrada posee de un subsistema para el envío de datos a través de algún protocolo de comunicación. El gestor de mantenimiento envía datos relevantes al sistema IoT para tenerlos en cuenta en el análisis de cuellos de botellas.	
Postcondición	Intercepción de datos, paso de datos a gestor comunicaciones y reenvío a capa de gestión de datos.	

Propósito
Recepción de datos de sistemas externos.
Resumen
Se reciben datos de las dependencias (Software de control principal estación, transporte entrada y gestor de mantenimiento), estos se interceptan y se validan que sean correctos para enviarlos a la gestión de datos, en caso contrario se reenvía un error a los sistemas origen.
Flujo Base
• Se interceptan los datos de envío por parte de los sistemas externos. • Se validan según unas reglas base con el tipo de dato correcto. • Se reenvía a la capa de Gestión de datos si validación OK. • Se reenvía un KO al sistema origen si no ha pasado la validación.

FU-COMO001: Funcionalidad principal para la gestión de comunicación.

- Extracción de datos MQTT.
- Extracción de datos Servicios REST.
- Extracción de datos transmisión Ethernet.

CU / Funcionalidad	Funcionalidad para la gestión de la comunicación	FU-COM001
Actores	Sistema IoT, sistemas externos, módulo de gestión de datos	
Tipo	Base	
Precondición	Hardware IoT envía los datos a través de servicios REST y Proxy ha interceptado los datos según el diagrama de comunicaciones del apartado 3.3.1.1. de este documento.	
Postcondición	Se han enviado los datos al gestor de datos	

Propósito
Gestionar los datos enviados a través de los diferentes protocolos de comunicación utilizados, principalmente MQTT, REST y FTP.
Resumen
Los sistemas hardware IoT y proxy con información de dependencias envían datos a través de los canales de comunicación, la interfaz de comunicación recibe los datos a través de la implantación de clientes de los protocolos indicados en el propósito MQTT, REST y FTP.
Flujo Base
Cliente servicio REST hardware IoT: • Se recogen los datos de los tres sensores enviados según la funcionalidad FU-CSA003 por el software del hardware IoT y se persisten en una BBDD del sistema. • La funcionalidad de Gestión de datos recoge el dato para persistirlo. Subscriptor MQTT: • Se recogen datos enviados desde el proxy en el que caso que existan publicados por parte del proxy siguiendo el protocolo MQTT publicador/Subscriptor y los recoge la funcionalidad de gestión de datos. • La funcionalidad de Gestión de datos recoge el dato para persistirlo. Cliente servicio REST proxy:

- Se recogen datos enviados desde el proxy (funcionalidad FU-IDD001) en el que caso que existan publicados por parte de Servicios REST y los recoge la funcionalidad de gestión de datos.
- La funcionalidad de Gestión de datos recoge el dato para persistirlo.

Cliente FTP proxy:

- Se recogen datos enviados desde el proxy en el que caso que existan enviados por parte del proxy por servicio FTP y los recoge la funcionalidad de gestión de datos.
- La funcionalidad de Gestión de datos recoge el dato para persistirlo.

3.1.1.4. FUNCIONALIDADES DE GESTIÓN DE DATOS

- FU-GDA001: Gestión de datos de dependencias externas y datos hardware IoT

CU / Funcionalidad	Funcionalidad que gestiona de datos de dependencias externas y datos hardware IoT	FU-GDA001
Actores	Módulo de gestión de comunicación	
Tipo	Base	
Precondición	Se ha realizado el tratamiento inicial por parte del módulo de comunicaciones según la descripción de la funcionalidad FU-COM001	
Postcondición	Se han gestionado y persistido los datos.	

Propósito
Extraer los datos adquiridos en el módulo de comunicaciones, transformarlo en el tipo de dato dentro del dominio del sistema y persistirlo
Resumen
El subsistema de gestión de datos recoge los datos extraídos de las dependencias externas desde el módulo de comunicación para proceder a su gestión, encapsulación en un objeto del dominio de negocio y persistencia del objeto.
Flujo Base
Flujo datos dependencias externas: • Se accede al dato de dependencias externas que ha sido extraído según la funcionalidad FU-COM001.

- Se encapsulan los datos en un objeto de dominio.
- Se persiste el objeto en BBDD

Flujo datos hardware iot:

- Se accede al dato procedente del hardware IoT que ha sido extraído según la funcionalidad FU-COM001.
- Se encapsulan los datos en un objeto de dominio.
- Se persiste el objeto en BBDD

- FU-GDA002: Funcionalidad de publicación de datos clave a través de microservicios.

CU / Funcionalidad	Funcionalidad de transferencia de datos clave a través de microservicios	FU-GDA002
Actores	Sistema de gestión de microservicios	
Tipo	Base	
Precondición	Se ha realizado el tratamiento inicial por parte del módulo de comunicaciones según la descripción de la funcionalidad FU-COM001	
Postcondición	Se han gestionado y publicado los datos en el microservicio.	

Propósito
Publicar los datos clave en un microservicio para que puedan ser consultados por clientes que necesiten manejar los datos clave relacionados con la detección de errores y cuellos de botella de la estación.
Resumen
Las áreas de producción acceden a los datos publicados por los microservicios para poder tener lectura de los datos clave.
Flujo Base

• Se construye la estructura de datos a publicar de las variables de interés de la estación de trabajo:
  ○ Cadencia de máquina
  ○ Detección envases entrada estación
  ○ Control/traza de botellas
  ○ Velocidad estación trabajo
  ○ Velocidad cinta entrada
  ○ Número de envases no aceptadas

○ Número de paquetes no aceptados ○ Listados de errores estación de trabajo • Se publican los datos en el microservicio, el cual se actualizará según la regla de tiempo que se determine (cada x minutos, por turno, por día, por lote, etc).

- FU-GDA003: Funcionalidad de tratamiento y carga de datos para reporte. Esta funcionalidad se podrá ver en el apartado específico de funcionalidad de reporte de datos.

3.1.1.5. FUNCIONALIDADES WEB Y NEGOCIO

- FU-WEB001 - Funcionalidad de Monitorización de datos clave/cuadro de mando cuello de botella estación de trabajo.

CU / Funcionalidad	Monitorización de datos clave / Cuadro de mando cuello de botella de estación de trabajo	FU-WEB001
Actores	Operarios (Monitorización), Usuarios responsables (Gestión)	
Tipo	Base	
Precondición		
Postcondición	Se han monitorizado los datos clave para verificar posibles errores en cuellos de botella.	

Propósito
Monitorizar datos clave de la estación de trabajo y su perímetro de control.
Resumen
El usuario revisa la interfaz de usuario sobre la monitorización de variables donde se podrán revisar cadencia, velocidad, gráfica en el tiempo de la variación de envases minuto, estadísticas de paradas, envases expulsados y otros indicadores importantes para detectar cuellos de botella. Los paneles emergentes ayudarán a informar de indicios de cuellos de botella.
Flujo Base
• El usuario accede a la interfaz de cuadro de mando • Visualiza los datos relevantes que permiten ver tres apartados principales: ○ Apartado de indicadores de cadencias (envases horas):

- Estación de trabajo
- Cinta de entrada
- Cinta de salida

- Apartado con un gráfico a lo largo del tiempo de la evolución de cadencia de envases hora en la estación de trabajo.
- Apartado con diferentes estadísticas:
  - Botellas expulsadas en última hora
  - Repeticiones parada estación > 5 minutos en última hora
  - Repeticiones parada transporte entrada > 5 minutos en última hora

En el apartado pantallas prototipo se podrá ver con detalle la interfaz de usuario ejemplar asociada a esta funcionalidad.

- FU-WEB002: Funcionalidad gestión de reportes. (Pantalla gestión de reportes)

CU / Funcionalidad	Gestión de reportes	FU-WEB002
Actores	Usuarios responsables (Gestión)	
Tipo	Base	
Precondición		
Postcondición	Se ha podido tramitar las gestiones de reporte	

Propósito
Gestionar la generación, consulta y envío de Informes con datos clave para tomar decisiones ante la detección de cuellos de botella
Resumen
El usuario accede a la funcionalidad de gestión de reportes de la interfaz Web para tramitar informes, la funcionalidad permitirá • Generar el informe a partir de un criterio base: ○ Datos actualizados en la última hora. • Consulta por pantalla del informe. • Envío a un repositorio de ficheros.
Flujo Base Generar: • Actor accede a la interfaz de usuario y pulsa opción de generador.

- El Sistema accede a la lógica de negocio encapsulada en la funcionalidad FU-RDA001.
- Se retorna el informe generado y guardado.

Consulta:

- Precondición: Informe generado previamente
- Actor accede a la interfaz de usuario y pula opción de consultar informe.
- El sistema recupera el elemento guardado.
- Se muestra por pantalla el informe generado.

Envío:

- Precondición: Informe generado previamente
- Actor accede a la interfaz de usuario y pula opción de consultar informe.
- El sistema recupera el elemento guardado.
- Se envía al repositorio de ficheros el informe generado.

- FU-WEB003: Gestión de datos históricos. (pantalla con opciones para seleccionar gráficos de históricos)

CU / Funcionalidad	Gestión de datos históricos	FU-WEB003
Actores	Usuarios responsables (Gestión)	
Tipo	Base	
Precondición		
Postcondición	Se han monitorizado gráficos asociados a historial de problemáticas de diferentes variables de la estación tratada.	

Propósito
Consultar los históricos de diferentes ítems del proceso y problemáticas a través de gráficos en un intervalo de tiempo definido por el usuario
Resumen
El usuario accede a la interfaz de la funcionalidad de gestión de datos históricos para seleccionar monitorizar en un rango alguna de las siguientes variables: • Cadencia de estación. • Botellas expulsadas. • Número de paradas de estación > 5 minutos • Número de paradas de transporte > 5 minutos • Frecuencia de problemáticas principales
Flujo Base

- El usuario accede a la interfaz de gestión de datos históricos
- Selecciona una fecha concreta
- Valida con el botón de generar gráfico
- Se visualiza el gráfico con las repeticiones

3.1.1.6. FUNCIONALIDADES REPORTE DE DATOS

FU-RDA001: Funcionalidad para la generación automática de reporte de datos a usuarios clave del sistema (incluye funcionalidad de la capa de gestión de datos, FU-GDA003: Funcionalidad de tratamiento y carga de datos para reporte).

¿Quién lo pide?

El reporte de datos se puede pedir de forma on-line a través de una función dentro de la interfaz de usuario Web o bien a través de una gestión automática que enviará los reportes cada intervalo de tiempo prefijado.

- La aplicación online, puede demandarlo a través de una función y se cogen los últimos datos guardados en la última hora. La función se compone de tres componentes:
  - Generar informe
  - Consultar informe
  - Enviar informe

- El sistema asíncrono generará informes cada intervalo de tiempo prefijado. Este tiempo puede ser parametrizado conforme la necesidad (cada hora, cambio de turno, diario, etc). Una vez generado se persistirá y enviará a través de un servidor de correo o depositado en un repositorio de ficheros

¿Por qué se pide?

- La funcionalidad on-line tiene sentido cuando se detectan niveles incorrectos en las variables clave o supervisión del proceso de envasado de la estación afectada.
- La modalidad off-line, tendrá sentido al hacer un informe por ejemplo diario para reportar de la situación en cuanto a problemas de cuellos de botella de la estación específica que está siendo monitorizada por el sistema IoT.

Necesidad de reglas de negocio para concretar cuando se produce un cuello de botella.

En un contexto real será necesario concretar una serie de reglas de negocio para determinar los índices por los cuales se debe mover la producción para que se detecten cuellos de botellas.

También para determinar un nivel de alerta de cuellos de botella, de esta forma se puede predecir si existe algún tipo de tendencia.

¿Qué contiene el ejemplar de reporte?

Datos relevantes que procedan a indicar un nivel de cuello de botella, a partir de la comparación de los índices en base a las reglas mencionadas previamente. Para el alcance de este trabajo y dado que no se centra en un dominio real sobre una línea de envasado de productos concreto (con valores pre-configurados), los datos serán simulados. Estos se podrán ver en el apartado "Simulación de situaciones" de este documento.

¿Cómo se procede?

Un proceso de tres fases:

- Extracción de datos. De la base de datos a través de una consulta.

- Tratamiento:
    - Se comparan los índices con rangos, máximos y mínimos.
    - Por cada ítem medido se adjunta información nivel del cuello botella
    - Se hace una valoración final:
        - Valor global que informará del nivel e impacto.
        - Recomendaciones en formato de acción.
- Guardado y envío de informe.
    - Se escribe el informe en un formato que sea claro, html.
    - Se persiste el informe bbdd.
    - Se envía a buzón de usuarios clave.

CU / Funcionalidad	Funcionalidad de tratamiento y carga de datos para reporte	FU-RDA001 FU-GDA003
Actores	Usuarios responsables (Gestión)	
Tipo	Base	
Precondición	Los datos clave del sistema están persistidos para poder preparar consultas y extracción de datos.	
Postcondición	Estructura de datos preparados para ser enviados a la funcionalidad de Reporte.	

Propósito
Generador automático que procesa datos clave de estación de trabajo y perímetro para generar un reporte con información sobre problemática detectada de cuello de botella.
Resumen
Esta funcionalidad podrá ser ejecutada desde la interfaz de usuario a través de la funcionalidad de reportes o bien ejecutada a través de la programación un proceso asíncrono. Las posibles situaciones son: Un usuario pide la generación de un informe desde la Interfaz Web de usuario. A partir de aquí, se procesará los datos de petición y los devolverá a la funcionalidad de reporte de datos para que informe con la respuesta. El sistema pide la ejecución de un informe de forma asíncrona a partir de la programación de un proceso por lotes la generación de un informe. A partir de aquí, se procesará los datos

de petición y los devolverá al job para ejecute la siguiente fase de envío con la respuesta de informe y lo deposite en el filesystem especificado por el sistema.
Flujo Base

• El sistema recibe la petición de reporte on-line o asíncrono.
• Se lanza el proceso un proceso de reporte:
  ○ Se realiza la consulta de los datos clave.
  ○ Se escriben sobre los objetos de dominio de reporte.
  ○ Se ejecuta comparación con índices (MAX, min) para determinar el nivel de problema existente de cuello de botella.
  ○ Se extrae la información que concreta cada nivel detectado.
  ○ Se escriben los índices y valoraciones.
  ○ Se guarda el reporte en el fileSystem.
  ○ Se guarda la ruta del reporte en BBDD.

Se muestra el diagrama de dominio base del sistema a alto nivel. En este se muestran las entidades que intervendrán en el sistema y la relación entre ellas:

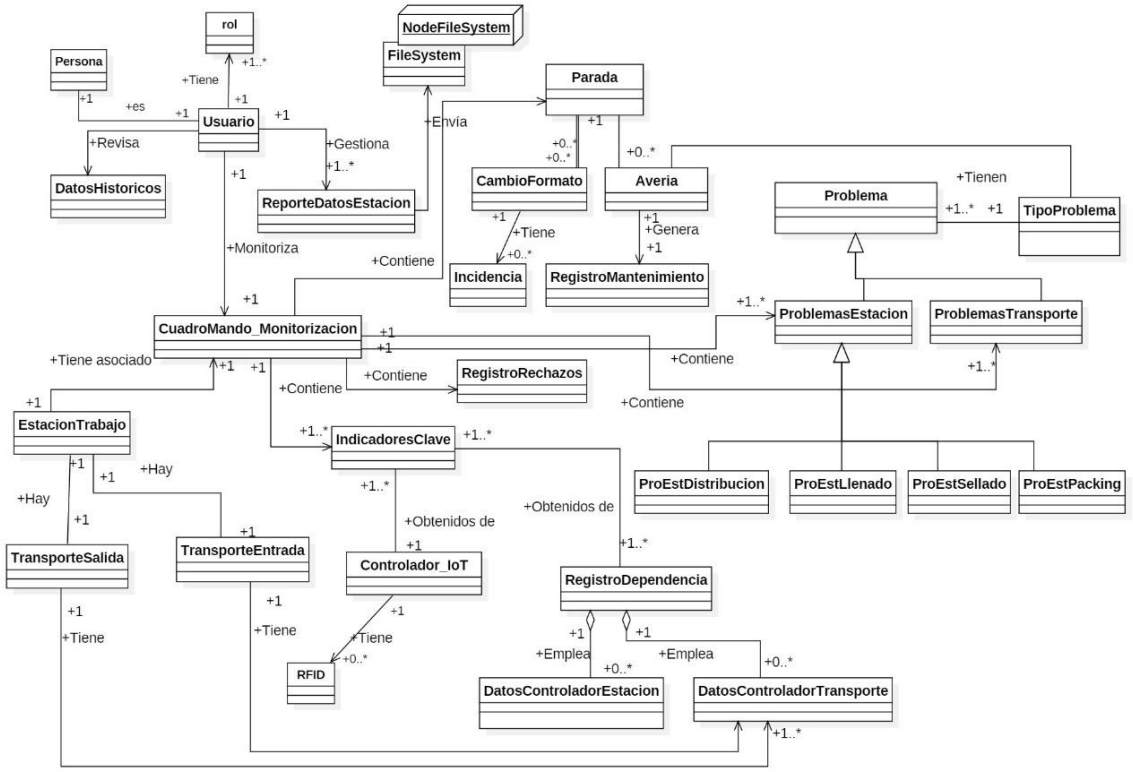


ILUSTRACIÓN 38: DIAGRAMA ESTÁTICO DE ENTIDADES

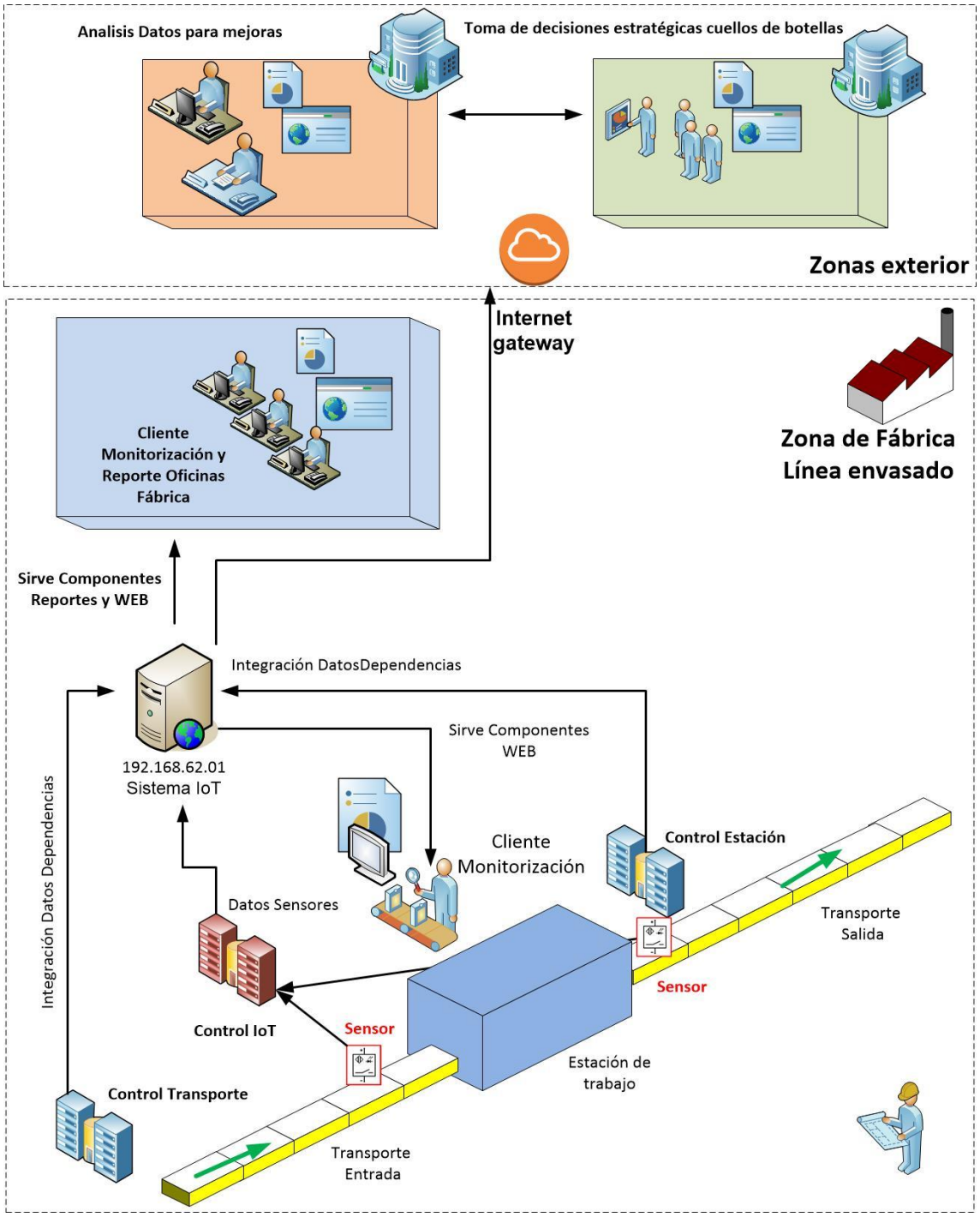


ILUSTRACIÓN 39: MODELO DE SISTEMA IOT A NIVEL DE ESTACIÓN

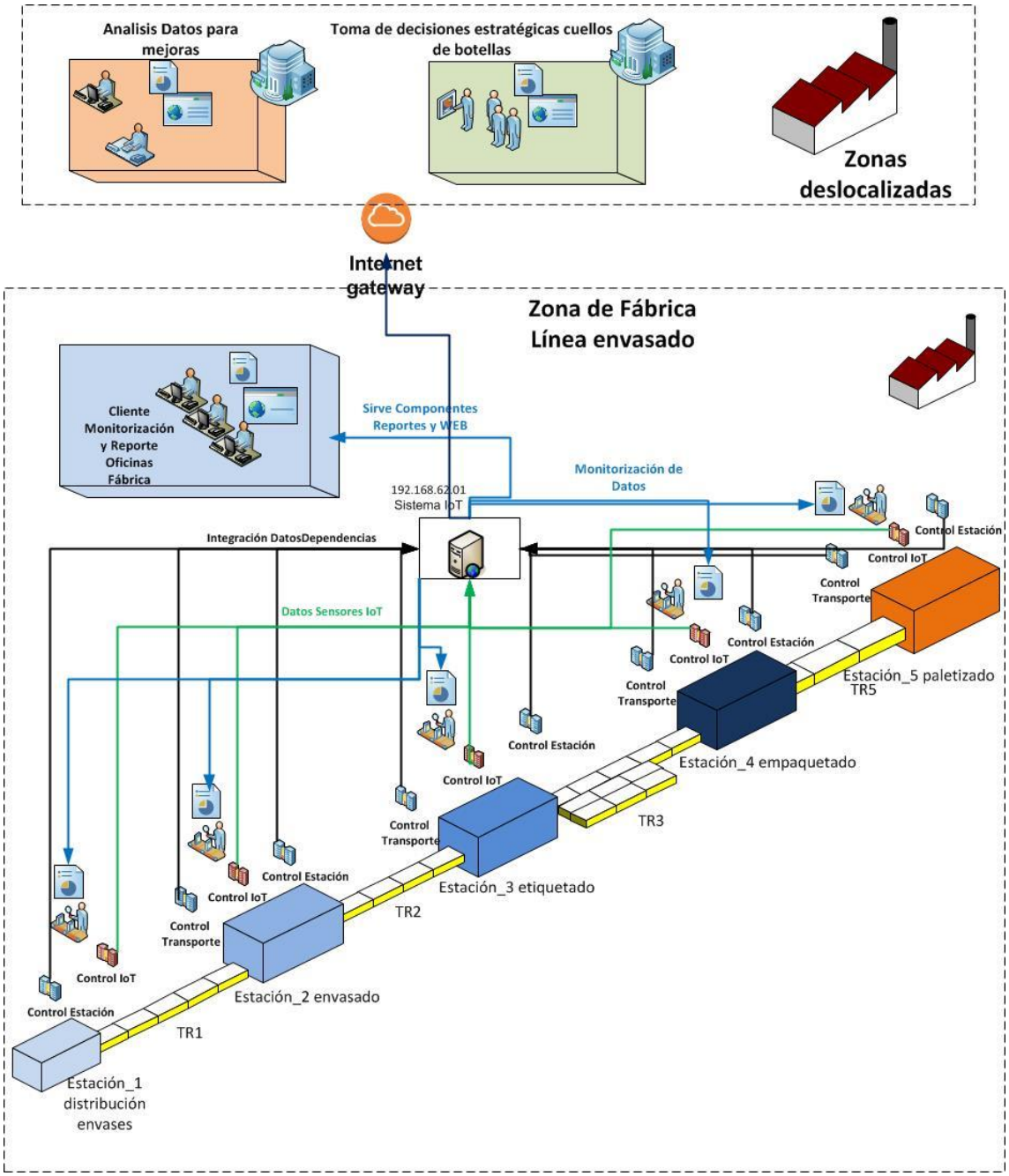


ILUSTRACIÓN 40: MODELO GLOBAL DEL SISTEMA IOT

3.1.1.8. PANTALLAS MAQUETA DEL PROTOTIPO

3.1.1.8.1. MONITORIZACIÓN DE DATOS

La siguiente maqueta muestra las líneas generales de un prototipo de pantalla donde se visualizan los datos clave para monitorizar un posible cuello de botella.

En ella se identifican tres trozos de pantalla principal:

- El primer tercio superior de pantalla se mostrarán cuatro valores principales, los tres primeros relacionados con la cadencia en envases hora en los transportes y estación y un último del número de paros en estación en el turno actual. Estos indicadores serán comparados con los valores óptimos para tener una referencia.
- El segundo tercio mostrará un gráfico para ver en las últimas 8 horas la evolución en la cadencia de estación de trabajo y poder comparar con el resto de estaciones.
- Por último, el último tercio muestra una estadística con problemáticas relacionadas con la estación de trabajo: Botellas expulsadas y paradas de más de 5 minutos producidas en estación y transporte de entrada.

El prototipo permitirá tener mensajes emergentes incluso sonoros para avisar de la extralimitación de valores en un determinado tiempo, como alarmas ante riesgo de cuello de botella.

En definitiva, El objetivo que se persigue esta monitorización es que el sistema diseñado muestre la situación actual (con un menor retraso respecto al tiempo real) mediante indicadores clave los indicios de cuello de botella a través de la supervisión y alarmas a través de mensajes emergentes. Un centinela de cuellos de botella.

ILUSTRACIÓN 41: PANTALLA MAQUETA MONITORIZACIÓN DATOS CLAVE PARA EVITAR CUELLOS DE BOTELLA

3.1.1.8.2. GESTIÓN DE REPORTE

La maqueta muestra un diseño de prototipo simple para poder generar un informe a partir de la selección del combo de opciones, estas podrán ser la última hora, turno actual y día actual.

Por otro lado, habrá una modalidad de poder consultar un informe por fecha.

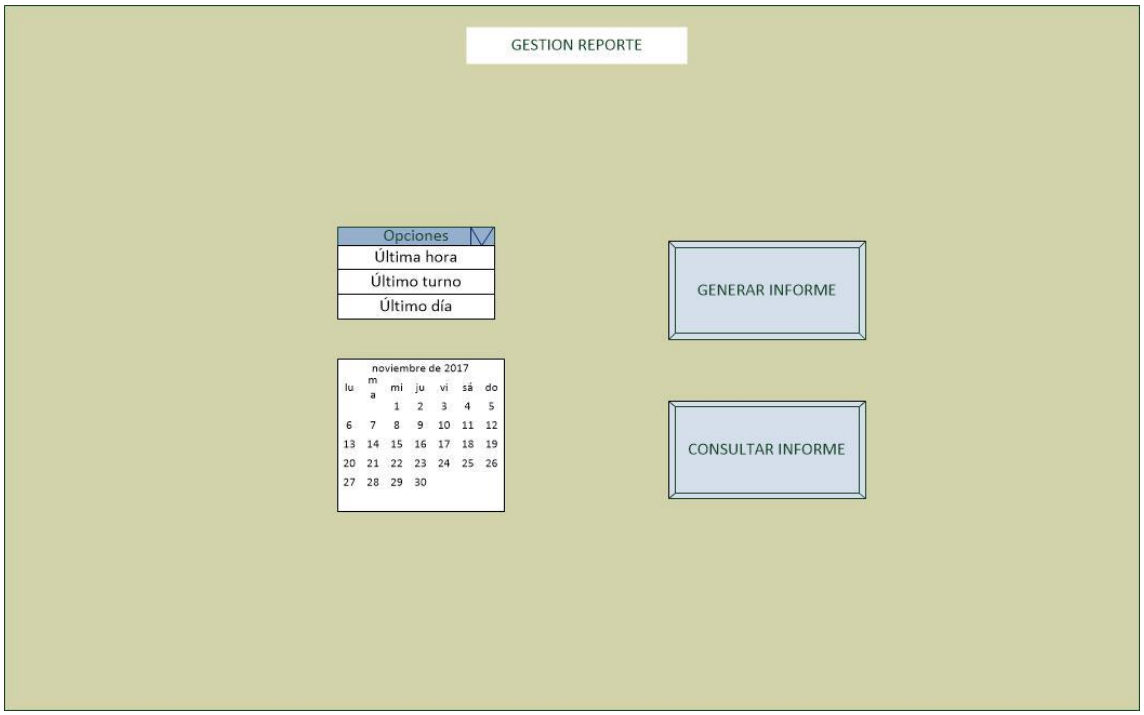
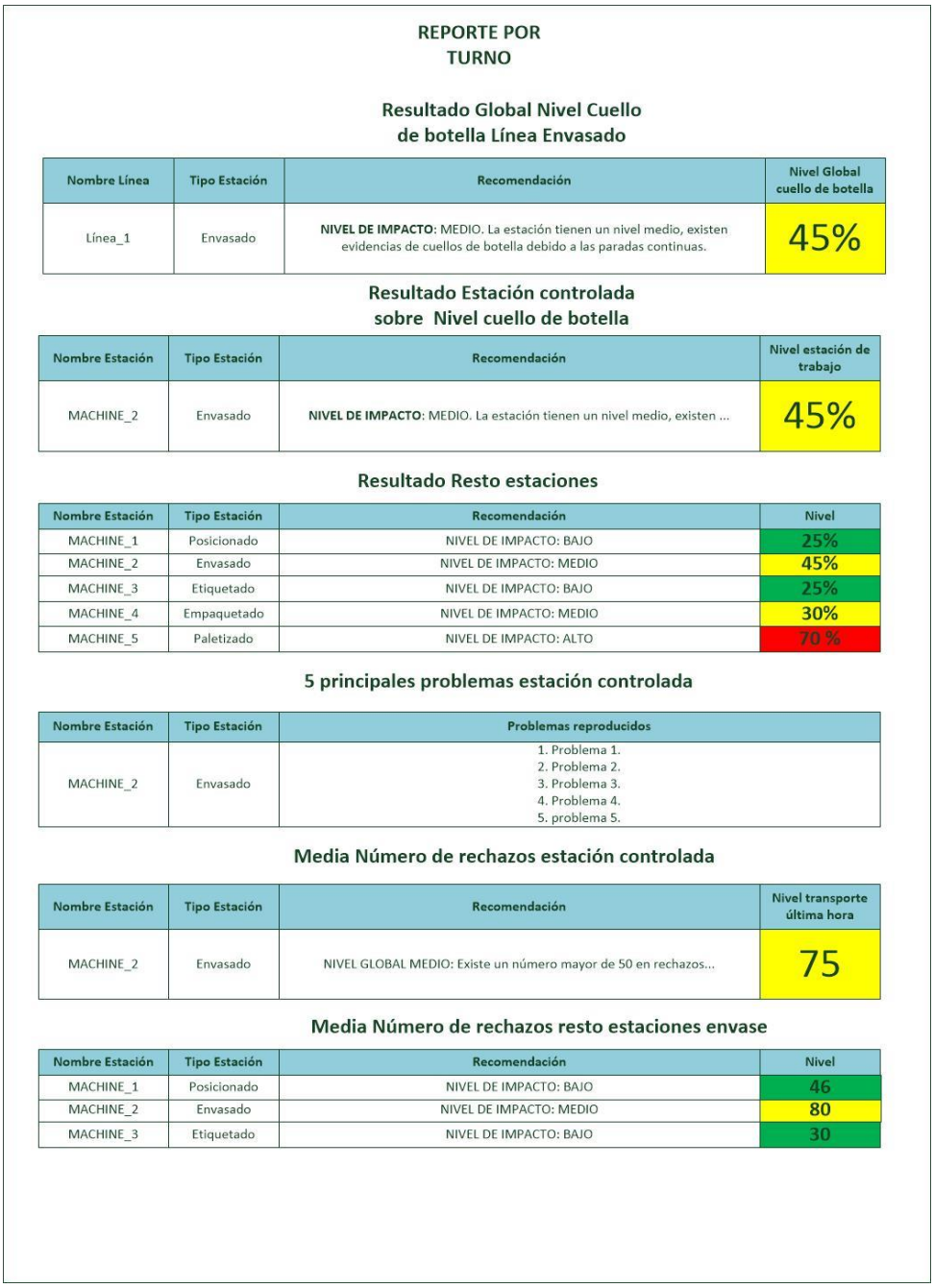


ILUSTRACIÓN 42: PANTALLA MAQUETA PARA GENERAR REPORTES ONLINE POR PANTALLA

Esta otra pantalla, muestra una maqueta con la simulación de un resultado de los datos obtenidos en un turno diario, el ejemplar muestra una serie de indicadores que detallan el resultado de un turno en cuanto a nivel de cuello de botella a nivel global de la línea de envasado y estación detectada y tratada por el sistema.



REPORTE POR TURNO

Resultado Global Nivel Cuello de botella Línea Envasado

Nombre Línea	Tipo Estación	Recomendación	Nivel Global cuello de botella
Línea_1	Envasado	**NIVEL DE IMPACTO**: MEDIO. La estación tienen un nivel medio, existen evidencias de cuellos de botella debido a las paradas continuas.	45%

Resultado Estación controlada sobre Nivel cuello de botella

Nombre Estación	Tipo Estación	Recomendación	Nivel estación de trabajo
MACHINE_2	Envasado	**NIVEL DE IMPACTO**: MEDIO. La estación tienen un nivel medio, existen ...	45%

Resultado Resto estaciones

Nombre Estación	Tipo Estación	Recomendación	Nivel
MACHINE_1	Posicionado	NIVEL DE IMPACTO: BAJO	25%
MACHINE_2	Envasado	NIVEL DE IMPACTO: MEDIO	45%
MACHINE_3	Etiquetado	NIVEL DE IMPACTO: BAJO	25%
MACHINE_4	Empaquetado	NIVEL DE IMPACTO: MEDIO	30%
MACHINE_5	Paletizado	NIVEL DE IMPACTO: ALTO	70 %

5 principales problemas estación controlada

Nombre Estación	Tipo Estación	Problemas reproducidos
MACHINE_2	Envasado	1. Problema 1. 2. Problema 2. 3. Problema 3. 4. Problema 4. 5. problema 5.

Media Número de rechazos estación controlada

Nombre Estación	Tipo Estación	Recomendación	Nivel transporte última hora
MACHINE_2	Envasado	NIVEL GLOBAL MEDIO: Existe un número mayor de 50 en rechazos...	75

Media Número de rechazos resto estaciones envase

Nombre Estación	Tipo Estación	Recomendación	Nivel
MACHINE_1	Posicionado	NIVEL DE IMPACTO: BAJO	46
MACHINE_2	Envasado	NIVEL DE IMPACTO: MEDIO	80
MACHINE_3	Etiquetado	NIVEL DE IMPACTO: BAJO	30

ILUSTRACIÓN 43: VISTA MAQUETA REPORTE CON INDICADORES Y RECOMENDACIONES DE PREVENCIÓN DE CUELLO DE BOTELLA

3.1.1.8.3. GESTIÓN HISTÓRICOS

En la siguiente maqueta muestra un formulario con diferentes posibilidades para generar gráficos de los indicadores clave según una fecha seleccionada:

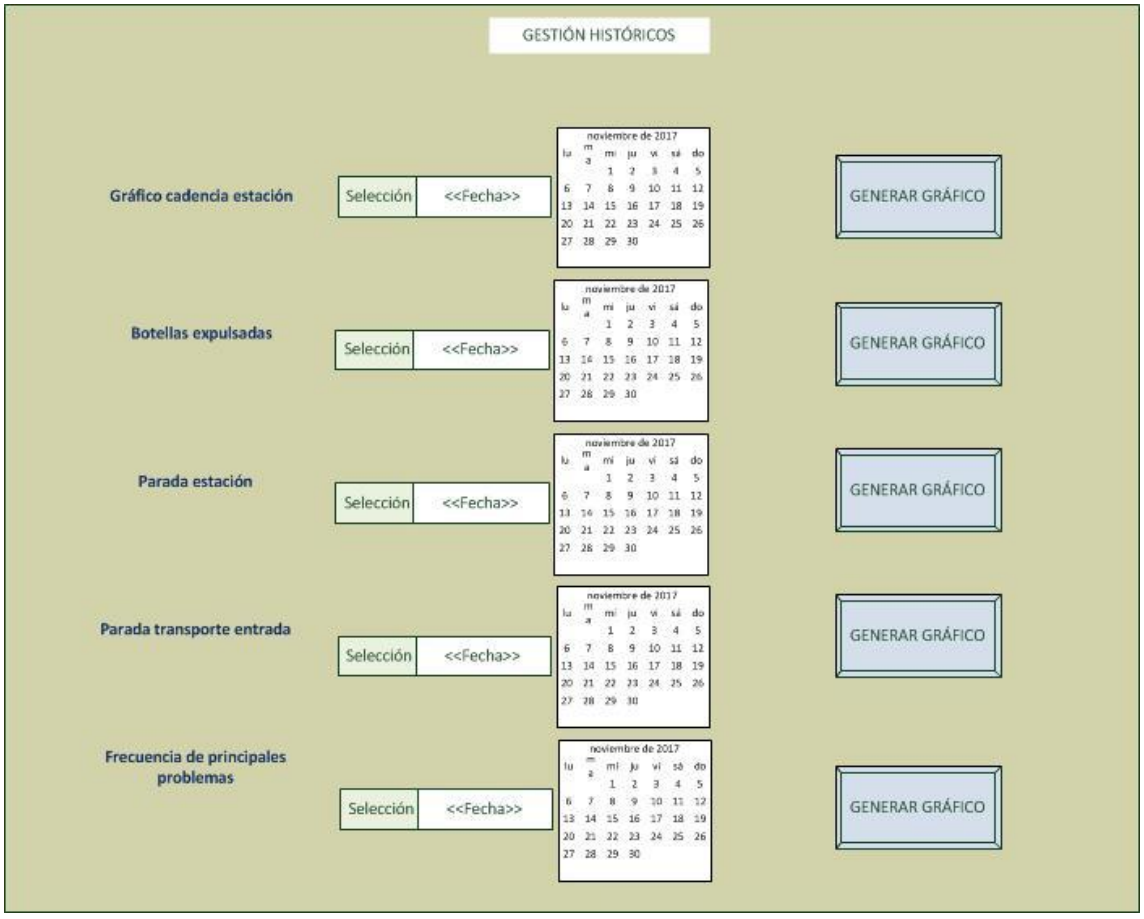


ILUSTRACIÓN 44: VISTA MAQUETA PARA CONSULTAR GRÁFICOS INDICADORES CLAVE SEGÚN FECHA

El objetivo de esta pantalla es conseguir un complemento para que el usuario pueda obtener datos de históricos y poder compararlos con los datos de la situación actual para así obtener un criterio sobre alguna problemática detectada de un posible indicio de cuello de botella.

Una vez pulsado “Generar gráfico” se mostrará un gráfico similar en aspecto como el siguiente ejemplo:

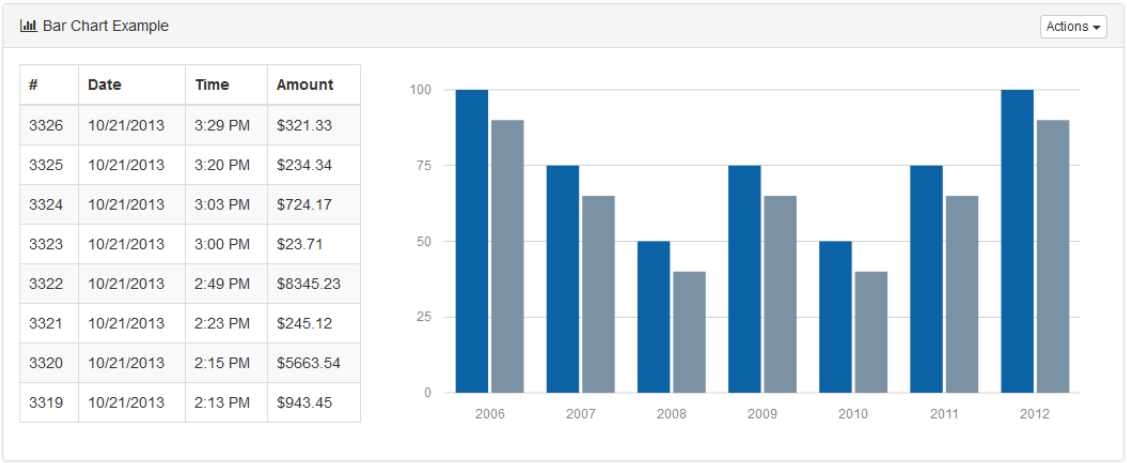


ILUSTRACIÓN 45: MODELO DE GRÁFICO PARA MOSTRAR LA TENDENCIA/EVOLUCIÓN DE UN INDICADOR CLAVE DEL SISTEMA.

3.1.2. SIMULACIÓN DE SITUACIONES

El siguiente apartado presenta un modelo de simulación con una serie de datos empíricos que modelan la información necesaria para poder generar un flujo de trabajo en todo el sistema.

Para poder simular una situación real, se partirá de la base del contexto definido en el capítulo 2.3 del apartado de investigación con una línea de envasado de 5 estaciones de trabajo y las siguientes reglas:

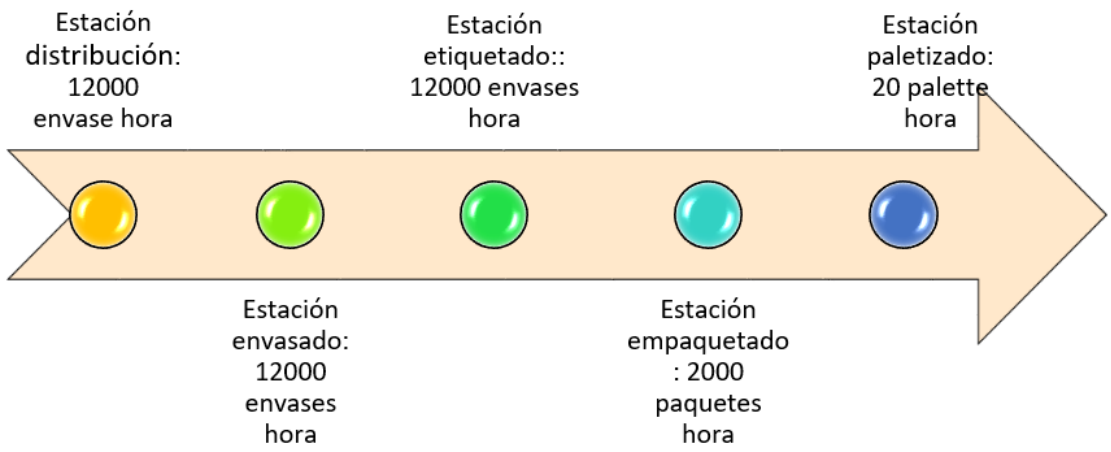


ILUSTRACIÓN 46: VALOR ÓPTIMO DE VALORES DE CADENCIA

El prototipo se centrará en el sistema IoT de una de las estaciones, en concreto de la estación de envasado, no obstante, se simulará la obtención de la comunicación del dato global de riesgo de cuello de botella recibido del resto de estaciones, para obtener una visión global.

Como resumen, en los siguientes subapartados se mostrarán valores máximos y mínimos a donde se tendrán que comparar las variables tanto para el proceso batch (por lotes) como para la monitorización online, datos que preceden de las dependencias, en concreto los datos de los controladores de gobierno del transporte de entrada y la estación de trabajo. Por último, los datos simulados en el controlador IoT.

Se clasifican los siguientes apartados con datos de simulación en las siguientes capas del modelo de sistema Iot:

- Simulación de datos del controlador IoT
- Simulación de datos de dependencias
- Reglas a nivel del sistema de monitorización y reporte.

3.1.2.1. TABLA DE HISTÓRICOS E ÍNDICES

Las tablas de históricos quedarán definidas en el modelo de datos en el apartado de especificación técnica de este documento.

3.1.2.2. SIMULACIÓN EN HARDWARE IOT

La funcionalidad FU-CSA002 descrita en el apartado 3.3.1.2 define el flujo base que se espera en cuanto a la detección de los tres sensores descritos.

Estos se resumían de la siguiente forma:

- *Tratamiento del sensor de cadencia (información del sensor de salida de estación)*
    - *Registra y acumula la detección de envases.*
    - *Controla el número de envase al minuto.*
    - *Realiza conversión a envases hora.*
    - *Prepara un registro de la cadencia registrada por minuto y se añade a un servicio Web para ser enviado.*

Para emular esta situación, se preparará una cadena de datos emulando variabilidad en cuanto a envases hora para ser enviados desde el dispositivo a cada minuto.

En el caso de una estación de envasado, el valor óptimo para la cadencia según el apartado de simulación de situaciones será de 12000 envases hora. Se escribirá un valor por cada minuto como el siguiente ejemplar:

Simulación de valor Cadencia = {9800, *yyyy-mm-dd hh:mm: ss}*

- *Tratamiento del sensor de entrada de estación RFID*
    - *Registra el identificador leído desde la antena RFID ante la detección de un envase nuevo.*
    - *Guarda el identificador en una tabla Hash con el momento de detección.*
    - *Prepara un vector con los registros adquiridos en un minuto y se añade a un servicio Web para ser enviado.*

Para emular este caso, se guardarán valores simulando el identificador detectado junto a la fecha y hora en un formato formal tal y como se muestra en la siguiente línea:

Simulación de valor Rfid = *{rfidvalue; yyyy-mm-dd hh:mm:ss}*

- *Tratamiento del sensor de entrada de estación*
    - *Controla la detección de envases nuevos.*
    - *Registra las repeticiones de detección entre botellas mayor a 10 segundos.*
    - *Registra tiempo medio entre botellas por minuto.*
    - Recoge los dos registros anteriores y se preparan para enviarlos por servicio Web a la capa superior.

Para emular este caso, se guardarán valores siguiendo la línea del siguiente ejemplar:

```
{"SensorEntrada": [ {
      "num_estacion": 1,
       "repiticionMayor10s": 4,
```

```
        “tiempoMedioEntreBotellas”: 5,
        "timestamp": 2018-11-10 12:34:12}]
}
```

Tabla resumen

Origen	Sensor	Frecuencia envío	Dato que se trata	Donde
Hardware IoT	Sensor cadencia	Cada minuto	Cadencia	Hardware IoT
			Momento detección	
Hardware IoT	Sensor RFID	Cada minuto	Valor identificación	
			Momento detección	
Hardware IoT	Sensor entrada estación	Cada minuto	Número estación	
			Número de veces que no ve envase mayor de 10 segundos.	
			Tiempo medio entre envases (sg)	
			Momento envío	

TABLA 4: TABLA DATOS HARDWARE IOT

3.1.2.3. SIMULACIÓN DE DATOS DE DEPENDENCIAS

Se simularán dependencias procedentes de tres fuentes distintas:

- Sistema de control principal Estación de trabajo.
- Sistema de control principal de transporte de entrada.
- Sistema de control del resto de estaciones de trabajo.

A partir de aquí, los datos que se simularán serán los siguientes:

- **Cadencia de transporte de entrada y estación**. Se simula una recepción de valores con el valor de la cadencia.

  Frecuencia de 5 minutos:

Simulación de cadencia estación = {Estacion_X, 9800, *yyyy-mm-dd hh:mm:ss}*
Simulación de cadencia transporte = {Cinta_X, 8900, *yyyy-mm-dd hh:mm:ss}*

- **Número de paradas mayores de 5 minutos por hora.**
  Frecuencia de 1 hora:

Simulación número paradas transporte > 5 minutos en 1 hora= {Transporte_X, num_paradas, *fecha_inicio_parada, tiempo_total_parada}*
Simulación número paradas estación > 5 minutos en 1 hora= {Estación_X, num_paradas, *fecha_inicio_parada, tiempo_total_parada}*

Regla de formato de fecha: *yyyy-mm-dd hh:mm:ss*

- **Número de envases rechazados.** Se parte de que cada estación de trabajo tiene un control de no aceptación, en una situación cada caso se concreta según sus reglas de aceptación como se muestran en los siguientes ejemplos:
  - Caso entrada estación etiquetado: Envases mal taponados.
  - Caso entrada estación empaquetado: Envases mal etiquetados

  Para emular la situación y acotarla a los datos de estación de envasado, se emula la recepción de datos.
  Frecuencia de 1 hora:

```
{"DatosEstaciones": {
      "estación_envasado": 1,
       "turno": lunes_mañana,
       “Numero_envases_KO”: 50,
       "timestamp": 2018-11-10 12:34:12}]
}
```

- Listados de errores que provienen del sistema de gestión de datos de mantenimiento de líneas y proceso industrial. Para emular esta situación se prepara una estructura como el siguiente ejemplar:

```
{"ListadoProblemas": {
“PRO001”: {
                “Origen”: Estacion_1
                “TipoProblema": Averia,
                “Descripción”: Rotura de muelles envasado,
                “Causa”: Mantenimiento incorrecto,
                "turno": lunes_mañana,
                “Nivel_impacto”: Muy alto,
                “Estacion”: 4,
                “Línea”: SERAC_2,
                "hora inicio": 2018-11-10 12:00:00,
                “hora fin”: 2018-11-10 12:16:00 }
“PRO002” : {
                “Origen”: Transporte_1
                “TipoProblema": Rendimiento,
                “Descripción”: Se observar lentitud inusual,
                “Causa”: Detecciones internas incorrectas,
                "turno": lunes_mañana,
                “Nivel_Impacto”: alto,
                “Estacion”: 2,
                “Línea”: SERAC_5,
                "hora inicio": 2018-11-10 13:00:00,
                “hora fin”: 2018-11-10 13:20:00 }
                ….

}
```

Tabla resumen

Origen	Simulación	Frecuencia envío	Dato que se trata	Donde
Controlador estación	Cadencia estación	Cada 5 minutos	Estación origen	**Se simula en capa de gestión de datos**
			Momento detección	
Controlador Transporte	Cadencia transporte	Cada 5 minutos	Transporte origen	
			Momento detección	
Controlador estación	Número de paradas > 5 minutos	Cada hora	Estación origen	
			Número paradas	
			Momento detección	
			Tiempo total parada 1 hora	
Controlador Transporte	Número de paradas > 5 minutos	Cada hora	Transporte origen	
			Número paradas	
			Momento detección	
			Tiempo total parada 1 hora	
Gestión de datos de mantenimiento de líneas y proceso industrial	Listado de problemas	Cada hora	Origen	
			Tipo de problema	
			Descripción	
			Causa	
			Turno	
			Nivel Impacto	
			Hora inicio	
			Hora fin	

TABLA 5: TABLA RESUMEN CON LA FRECUENCIA DE ACTUALIZACIÓN DE DATOS

3.1.2.4. SIMULACIÓN A NIVEL DE MONITORIZACIÓN Y REPORTE

A continuación, se muestran una serie de reglas a alto nivel que simulan una situación donde se puede generar cuello de botella, las cuales se tendrán en

cuenta para el reporte y los mensajes emergentes en la monitorización Web del sistema IoT.

Reglas a nivel monitorización y reporte

Según las funcionalidades descritas en el apartado 3.3.1.7 existían dos formas de emitir el informe, online y batch. Como regla de simulación, el reporte se limitará a los siguientes aspectos:

- Reporte Online:
    - Datos de Diario
    - Datos de Último Turno
    - Datos Última hora
- Reporte Batch:
    - Diario

Niveles

Todas las estaciones envases (distribución, envasado, llenado):

- Óptima: A partir de 12000 Envases hora
- Nivel cuello bajo >66%: Entre 7900 y 12000 Envases hora.
- Nivel cuello botella <66%: Por debajo de 7900 Envases hora.

Estación de empaquetado:

- Óptima: A partir de 2000 paquetes de 6 envases hora.
- Nivel cuello bajo >66%: Entre 1320 y 2000 paquetes hora.
- Nivel cuello botella <66%: Por debajo de 1320 paquetes hora.

Niveles establecidos por el número de rechazos (distribución, envasado, llenado):

- Óptimo: < 50 envases hora
- Nivel bajo: Entre 50 y 250 envases hora
- Nivel alto: > 250 envases hora

En las siguientes líneas, se muestra a alto nivel tipologías de problemas que se tendrán en cuenta para llevar a cabo la simulación y que en un entorno real ayudarán a determinar factores de riesgo de un cuello de botella.

Listado Problemas Estación

- Rechazos envases internos en intervalos de tiempo.
- Detalle problemas de Paradas por avería.
- Detalle problemas de Paradas no provocadas por avería (cambio de formato, etc).
- Problemas externos a estación.
- Tiempo improductivo. (tiempo de paradas)

Listado Problemas transporte de entrada

- Rechazos envases.
- Detalle problemas de Parada por avería
- Tiempo improductivo (tiempo de paradas)

Intervalo de tiempo del reporte indicadores de cuello de botella

Los intervalos de tiempo para manejar los reportes de datos en la simulación se realizarán en turnos simulados de 8 horas. A través de los cuales se puede tener en cuenta la variabilidad del proceso en respecto al factor humano del cambio de manejo de estación de trabajo sobre una situación real.

3.2. ESPECIFICACIÓN TÉCNICA

3.2.1. COMPONENTES DE ARQUITECTURA DE INFRAESTRUCTURAS Y SISTEMAS

3.2.1.1. ARQUITECTURA DE SISTEMAS E INFRAESTRUCTURAS

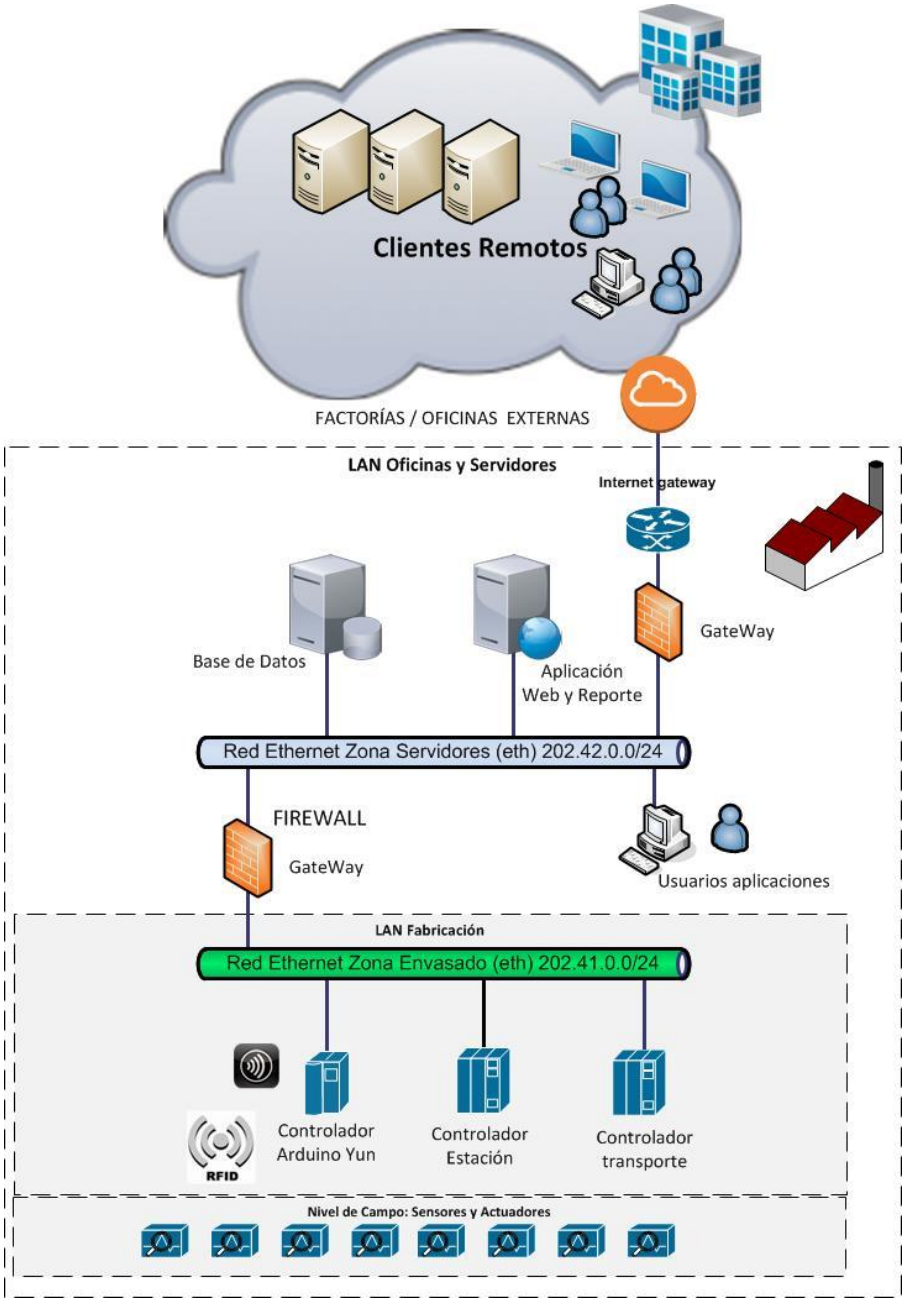


ILUSTRACIÓN 47: CONTEXTO DE INFRAESTRUCTURAS

Se resumen un contexto de infraestructuras a alto nivel, partiendo de la base que el sistema IoT residirá en un Centro de procesamiento de datos a nivel de línea de envasado.

En un contexto real –fuera del alcance de este trabajo-, se incluirán detalles a nivel de segmento de red, seguridad perimetral y exterior y otros elementos necesarios para cubrir toda una arquitectura de infraestructura que permita ofrecer servicio al sistema.

En la ilustración se pueden observar las distintas capas a alto nivel en el que se situarían los componentes a nivel de red.

En primer lugar, estarían los sensores y actuadores a un nivel de campo, dependiendo de la tecnología de conexión, se conectarán directamente al interfaz de entradas salidas de los dispositivos de control o bien utilizando un BUS de campo.

El controlador iot basado en Arduino, poseerá conectividad inalámbrica y adaptador para adquirir datos por RFID.

En la capa inmediatamente superior, se encontrarán los propios controladores, estos irán conectados a un segmento de red a nivel de fabricación donde se verán entre sí los distintos controladores de estaciones y transporte.

Un Gateway permitirá acceder a otro segmento de red donde se situará el servidor que acoge la aplicación de reporte y monitorización, desde este mismo segmento podrán acceder los usuarios de línea y oficinas a la monitorización y reporte.

En este mismo contexto, se introduce otra puerta de enlace a diferencia de fábrica el enlace será a través de la red global para acceder a los datos de fábrica desde las oficinas remotas.

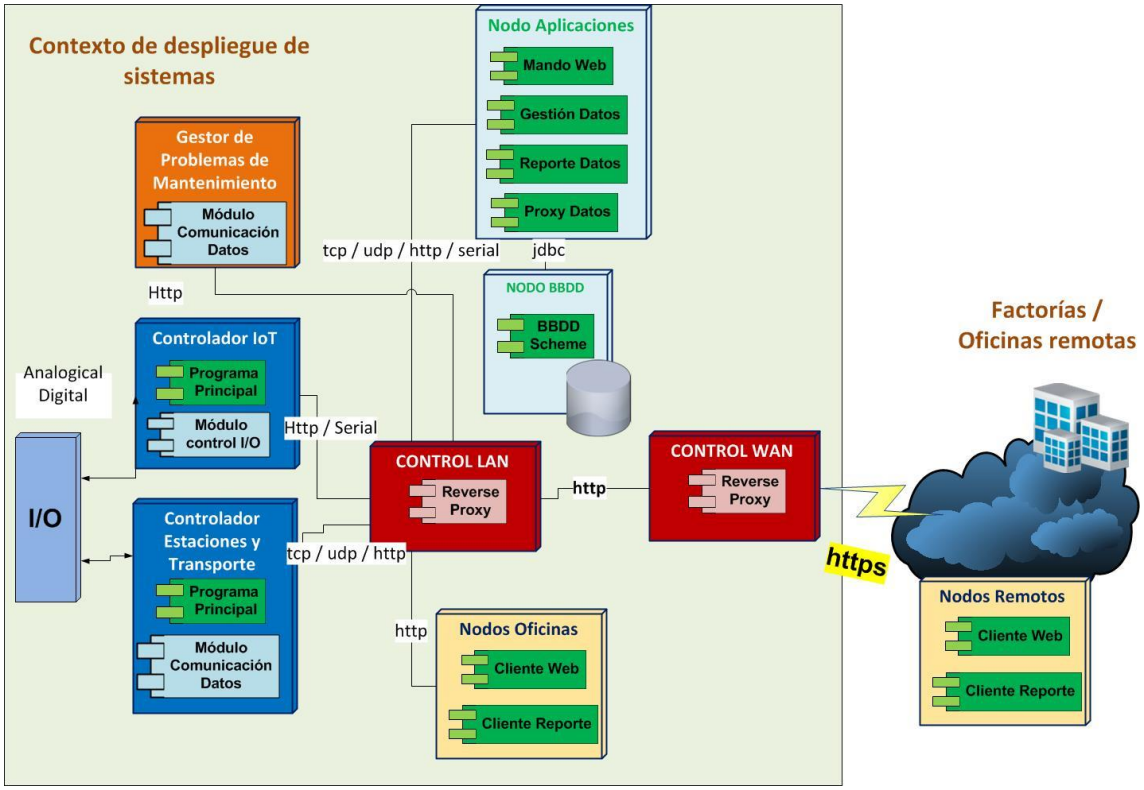


ILUSTRACIÓN 48: CONTEXTO DE DESPLIEGUE DEL SISTEMA

Alineado al contexto de infraestructuras ideado, la siguiente ilustración muestra una iniciativa de diagrama de despliegue de sistemas y un resumen de los principales componentes y la comunicación entre ellos.

A nivel de detección, se ha contemplado que el sistema adquiera datos con conexiones analógicas y digitales, sin tener en cuenta ningún bus de campo específico.

Los componentes del controlador IoT se comunicarán al contenedor de aplicaciones a través de comunicación serie y http, esta última vía servicios web. Los controladores de estaciones y transporte o mencionados funcionalmente como dependencias externas se comunicarán punto a punto tal y como se indicó en el modelo global de componentes, para ello se utilizarán los protocolos a nivel de transporte tcp y udp y a nivel de aplicación http

siempre y cuando el controlador permita un intercambio de datos vía web service u otra vía que utilice el protocolo http.

Se intercalan dos controladores de red, los dos poseerán proxys para controlar el tráfico interno y externo hacia los clientes de oficinas deslocalizadas.

El sistema IoT diseñado estará instalado en el contenedor de aplicaciones junto a la BBDD, en él deben de pasar filtrados los mensajes que proceden de los controladores.

Se incluye un nodo con un gestor de mantenimiento, este sería el elemento que comunica los datos relacionados con problemáticas, paradas, averías de estaciones de trabajo.

3.2.2. COMPONENTES DE ARQUITECTURA SOFTWARE

3.2.2.1. ARQUITECTURA LOGICA DEL SISTEMA

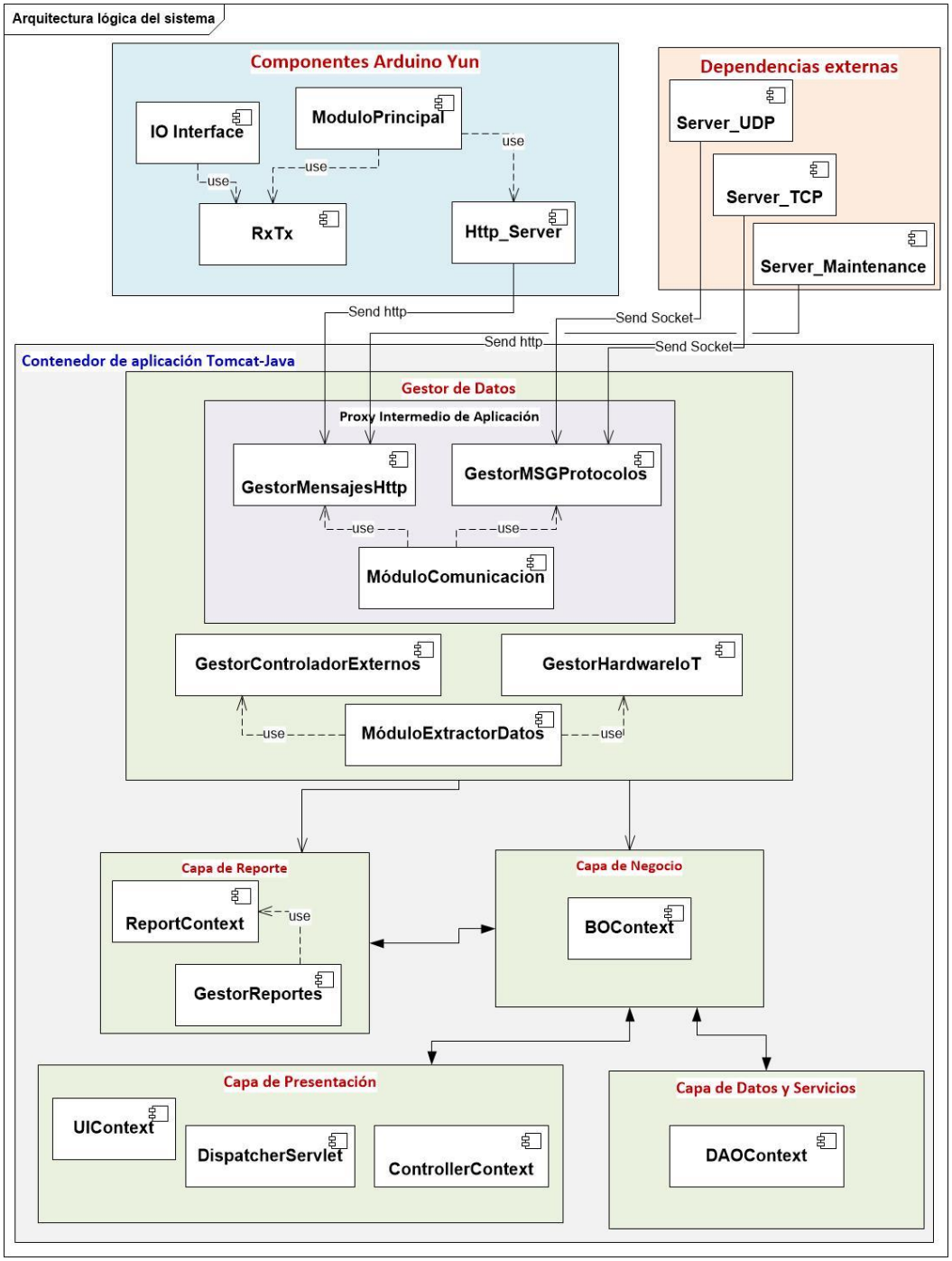


ILUSTRACIÓN 49: ARQUITECTURA LÓGICA DEL SISTEMA

Se definen tres bloques principales donde se cumplirá la siguiente lógica dentro del sistema:

- Componentes Arduino YUN: básicamente utiliza dos componentes, el programa principal donde se trata la información de los sensores y el servidor Web desde donde se emite el servicio REST con la información definida en la funcionalidad FU-CSA002 (Apartado 3.3.1.2 de este documento).
- Dependencias externas: Son los sistemas externos al contenedor de la aplicación los cuales emiten la información al sistema IoT. Se parte de la base que existirán envíos TCP y UDP (Controlador estación de trabajo y transporte) y otros vía Web (Sistema de gestión de mantenimiento de averías). Implementa la parte de comunicación externa en la funcionalidad descrita en FU-IDD001 (Apartado 3.3.1.3 de este documento).
- Contenedor de aplicación: Basado en un contenedor Tomcat bajo lenguaje Java, en este existirán los siguientes componentes para cubrir las funcionalidades definidas:
    - Gestor de datos:
        - Contendrá en un módulo aparte el proxy de aplicación con la lógica de intercepción de mensajes de Arduino Yun y Dependencias externas.
        - Contendrá la lógica para extraer transformar y almacenar los datos externos al contenedor y carga en otros casos si se precisa.

    - Capa de reporte:
        - Encapsulará la lógica para generar los reportes, las peticiones pueden venir desde la funcionalidad online de gestión de reporte o el sistema asíncrono para emitir reportes que en forma de simulación será cada 8 horas.

- Capa de Datos y Servicios:
  - Se encapsulará toda la lógica relacionada con el mantenimiento de BBDD y acciones de alta, actualización y borrado. La capa de Servicios también encapsula la lógica de envío de datos clave sobre los cuellos de botella en la estación controlada descritos en la funcionalidad. FU-GDA002.
- Capa de negocio: En la presente capa se implementará la lógica de negocio que encapsula las funcionalidades Web en las que se monitorizarán los datos clave del cuello de botella, gestión de reportes y de datos históricos.
- Capa de presentación: Los componentes resuelven el control y contexto de interfaz de usuarios a partir de la implementación del patrón Modelo Vista Controlador.

3.2.2.2. ARQUITECTURA DEL CONTROLADOR IOT

El Hardware mencionado como controlador IoT a lo largo del documento será un componente electrónico de control de código abierto, entre todos se escoge para este diseño un Arduino YUN [ARD01]. Esto es porqué posee capacidad de microcontrol para entradas y salidas digitales e incorpora un microprocesador que tiene la capacidad de usar servidores web además de poseer una interfaz de red inalámbrica por Wi-Fi y ethernet.

En la siguiente figura se puede ver el detalle de la segmentación del dispositivo, en el segmento de la izquierda está el microcontrolador que cuenta con una interfaz de entradas/salidas digitales y analógicas, mientras que el segmento de la derecha muestra el microprocesador que funciona bajo un sistema operativo basado en Linux:

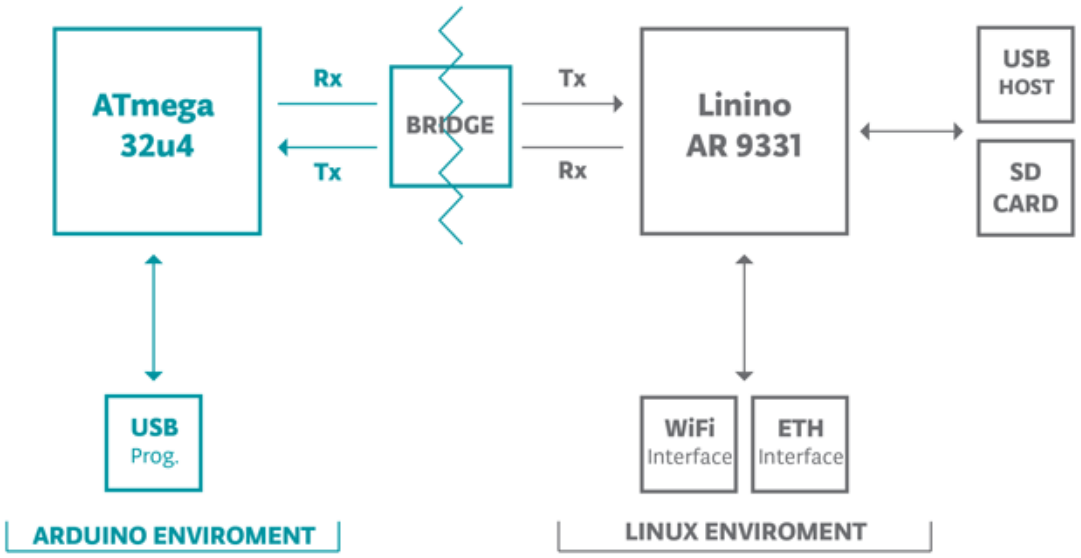


ILUSTRACIÓN 50: ARQUITECTURA BLOQUES ARDUINO YUN

EL Detalle de los esquemas lógicos de cada uno de los subcomponentes del sistema se muestran en https://www.arduino.cc/en/uploads/Main/YUN-V04(20150114).pdf

Por lo tanto, los componentes de este modelo cubren la funcionalidad de poder tratar y enviar vía Web Service la información a la capa de gestión de datos del sistema.

De la misma forma, cumple la premisa de ser un componente que acerca la información de sensores a internet, utilizado por lo tanto en el paradigma de IoT.

Para poder detectar envases se utilizarán las tecnologías mostradas en el Anexo 4 de este documento, en concreto sensores sin contacto físico. Concretamente, los válidos serán las sensores fotoeléctricos, capacitivos y ultrasónicos que hacen fluctuar la señal de entrada al dispositivo microcontrolador al paso de un envase.

En la aplicación dentro de los casos industriales de distintos formatos, jugará la investigación en cuanto a escoger el elemento más preciso y con menos tolerancia de errores, por ejemplo, ante botellas transparentes una célula fotoeléctrica que actúa al corte de un haz de luz puede tener mayor tolerancia

de errores por la transparencia mientras que un sensor ultrasónico será más preciso en este caso dado que detecta de forma invisible al contrario que la célula de haz de luz.

Como premisa, en el caso de la detección por RIFD, los envases deberán de llevar incrustados una etiquetad pasiva siguiendo el principio indicado para esta modalidad en el Anexo 5 de este documento.

Para este modo de trabajo de RFID solo se captará un código que identifica individualmente al envase. El sistema lo asociará los datos de hora de detección y turno. Estos datos podrán consultar el número de Lote de trabajo.

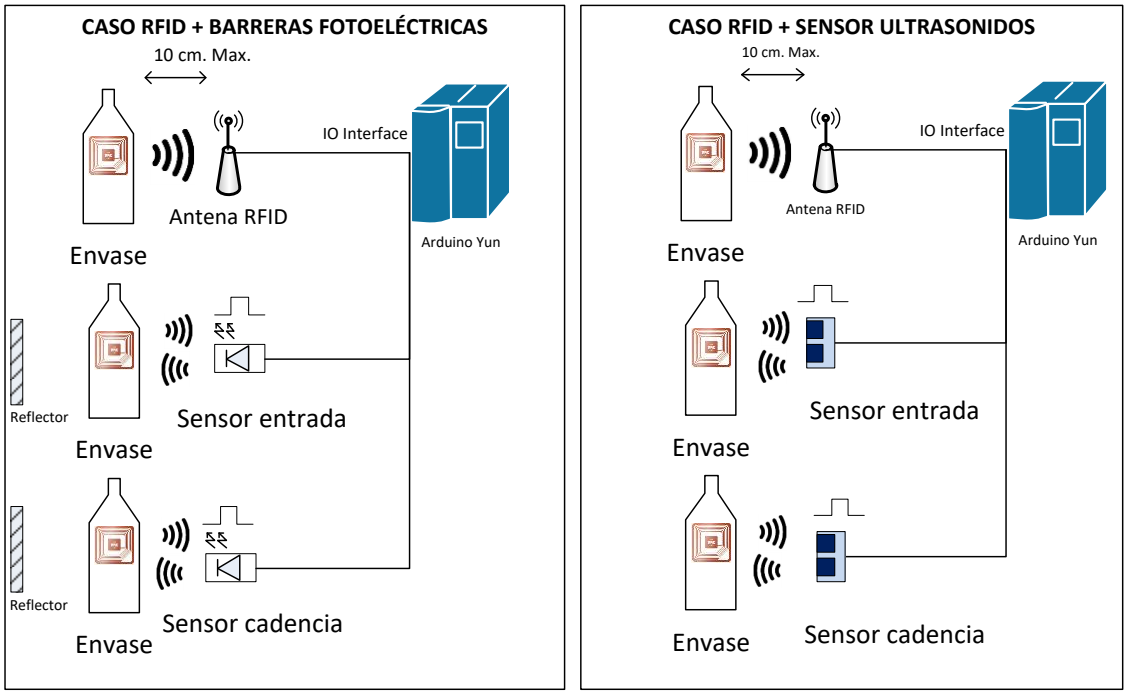


ILUSTRACIÓN 51: CONTRASTE ENTRE LA INCORPORACIÓN DE CÉLULAS FOTOELÉTRICAS Y SENSORES ULTRASÓNICOS EN EL DISEÑO.

El gráfico de la izquierda muestra un diseño con sensor RFID y detectores de entrada de envases y cadencia con sensores fotoeléctricos y reflectantes para devolver señal de detección. A la izquierda, se contará con sensores ultrasónicos para los detectores de entrada envases y cadencia.

Detalle de conexión del sensor RFID hacia Arduino:

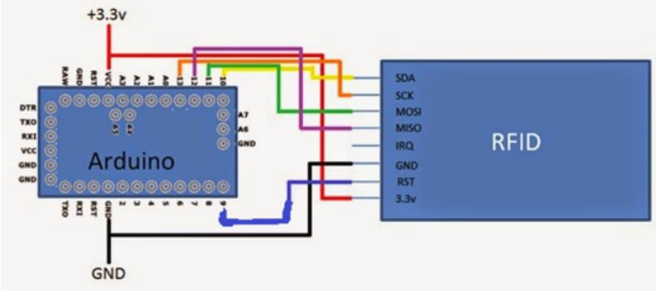


ILUSTRACIÓN 52: BLOQUE DE CONEXIÓN RFID SOBRE ARDUINO YUN

Detalle de conexión de un sensor de ultrasonido modelo HC-SR04:

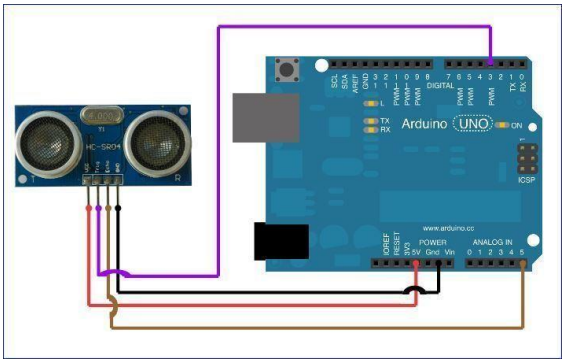


ILUSTRACIÓN 53: CONEXIÓN SENSOR ULTRASÓNICO HC-SR04 A ARDUINO

El siguiente modelo (mic+340/IU/TC) cuenta con una mayor precisión en la detección [MIC01]:

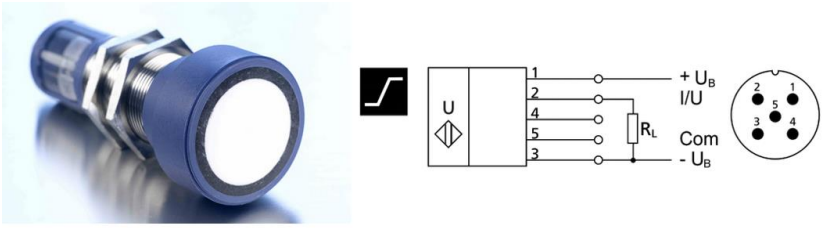


ILUSTRACIÓN 54: CONEXIÓN SENSOR ULTRASÓNICO MIC+340

Arquitectura de componentes de software:

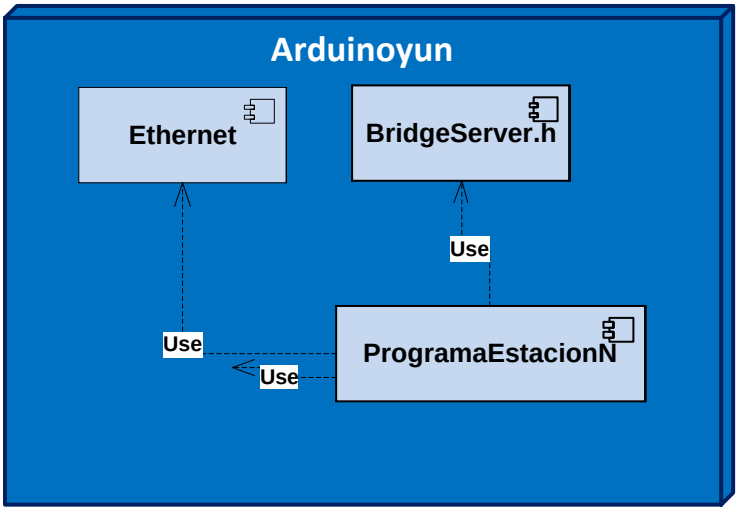


ILUSTRACIÓN 55: ARQUITECTURA COMPONENTES ARDUINO YUN

El controlador asociado a la estación de trabajo tendrá un programa principal que utilizará las librerías:

- BridgeServer: Esta librería servirá para conectar en su propio hardware con el microprocesador donde va alojado el contexto Linux y desde donde se invocarán las librerías para generar un servidor Web para que pueda enviar los datos a través de un servicio WEB tipo REST.
- Ethernet: Permitirá al dispositivo utilizar los puertos para poder enviar vía http el mensaje REST.
- ProgramaEstacionN: Contendrá el programa principal para implementar las funcionalidades descritas en el apartado 3.3.1.2 “Funcionalidades control sensor actuador IoT” de este documento.

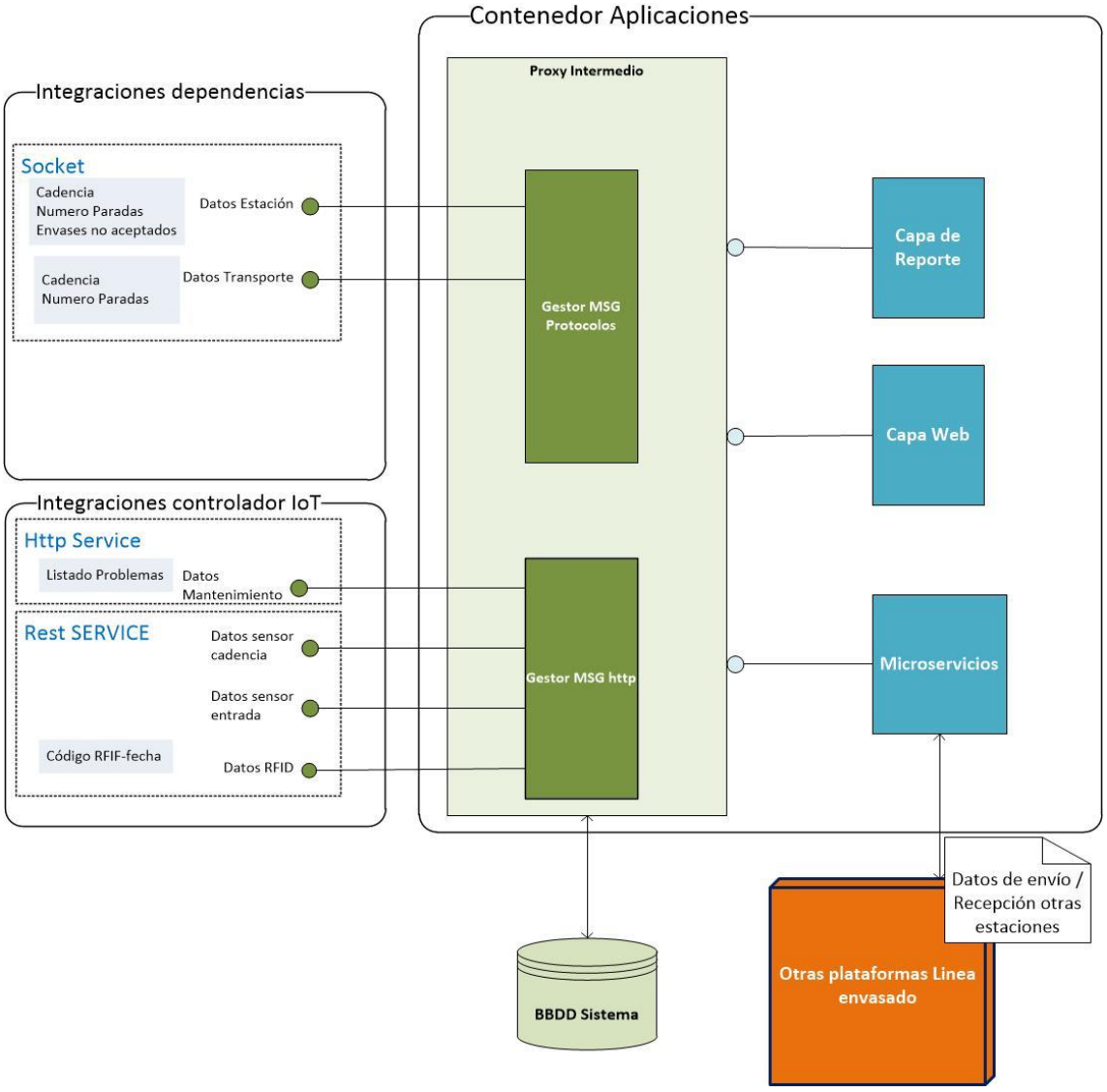


ILUSTRACIÓN 56: ARQUITECTURA INTEGRACIÓN DE DATOS

EL diagrama muestra la estructura que seguirá el sistema de integración de los datos que provienen tanto del controlador IoT como de los datos de dependencias.

El gestor de Datos contendrá la lógica de integración dos componentes, el proxy intermedio para absorber los datos que provienen de dependencias y un cliente WS para absorber los datos en formato WS del controlador IoT.

Habrá tres componentes consumidores de datos: La capa de reporte, la Web y los microservicios descritos en la funcionalidad FU-GDA002 del apartado 3.3.1.5 de este documento para publicar los datos para que sean absorbidos por clientes y el resto de estaciones de la línea de envasado. Como recordatorio estos serán los siguientes:

- *Se construye la estructura de datos a publicar de las variables de interés de la estación de trabajo:*
    - *Cadencia de máquina*
    - *Detección envases entrada estación*
    - *Control/traza de botellas*
    - *Velocidad estación trabajo*
    - *Velocidad cinta entrada*
    - *Número de envases no aceptadas*
    - *Número de paquetes no aceptados*
    - *Listados de errores estación de trabajo*

El sistema podrá persistir los datos para tener un registro de los datos llegados de los componentes externos y como elemento de capa de datos para poder realizar un tratamiento en cualquier momento.

3.2.2.4. ARQUITECTURA DEL DISEÑO WEB

El modelo que se seguirá para la arquitectura en la parte WEB del sistema es el patrón Modelo Vista Controlador y una estructura de 3 capas para tener un orden y estructura de la información capaz de organizar y orquestar el buen funcionamiento interfaces de usuario dentro del paradigma de la tecnología Web actual.

En primer lugar, el patrón de diseño modelo vista controlador define la estructura lógica que separa los datos y la lógica de negocio de la interfaz de usuario.

La secuencia de pasos se resume de la siguiente forma:

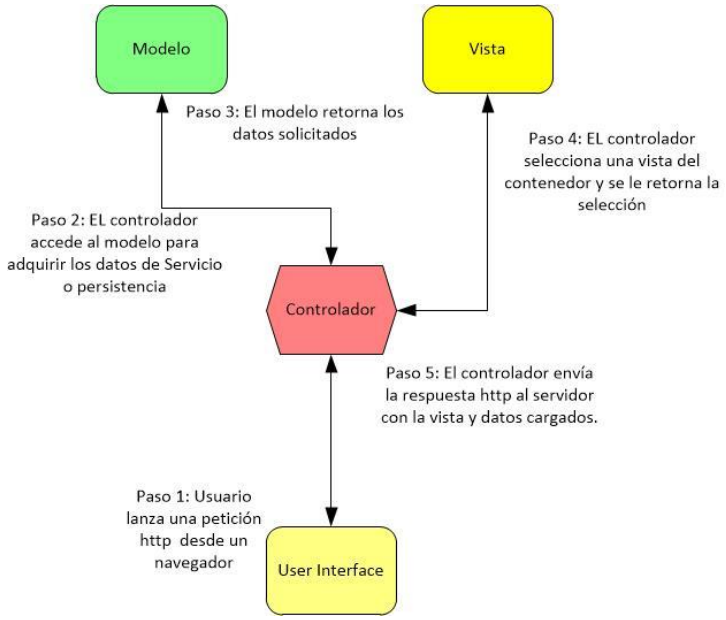


ILUSTRACIÓN 57: ESQUEMA DEL PATRÓN DE DISEÑO MVC

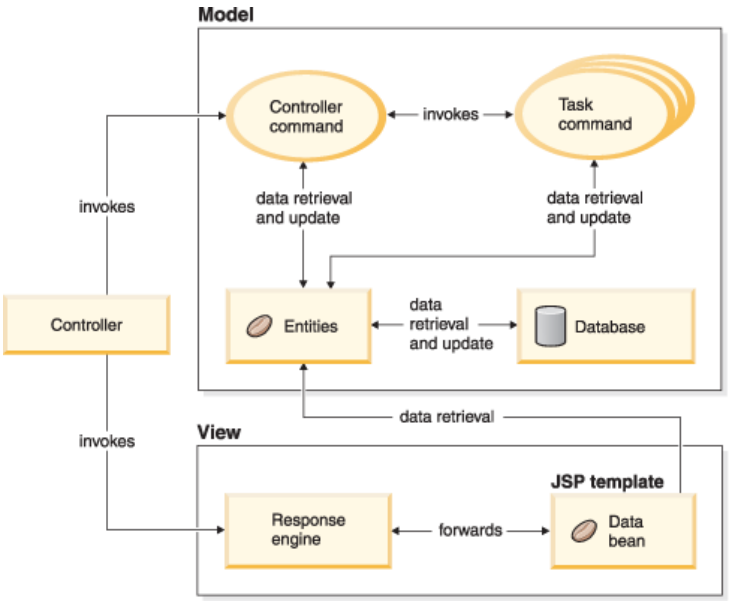


ILUSTRACIÓN 58: SECUENCIA DE PASOS DEL MODELO VISTA CONTROLADOR

Por otro lado, la estructura organizativa de la aplicación WEB en tres capas permite desacoplar los elementos y que cada contexto obtenga una responsabilidad.

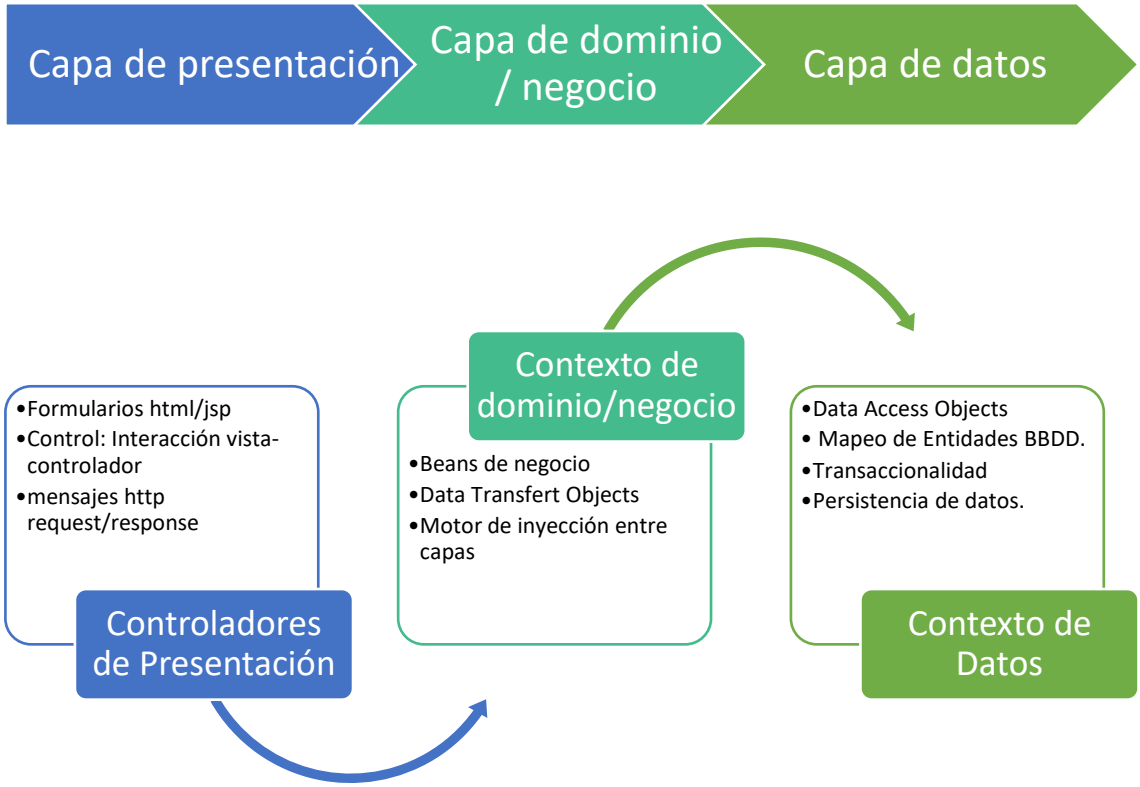


ILUSTRACIÓN 59: BLOQUES DE PATRÓN DE DISEÑO DE PROGRAMACIÓN EN 3 CAPAS

De forma indirecta intervendrán los siguientes patrones:

Singleton: Para poder limitar la creación de un solo acceso global a una clase a partir de un objeto único, será útil para tener una instancia del controlador en cada uno de los contextos y evitar una instancia nueva en cada método.

Fachada: Para mejorar la división entre capas y la inyección de componentes entre los diferentes contextos, pero manteniendo la independencia de cada capa se utilizará la estructuración a través del patrón fachada.

Como paradigma de programación se utilizará la orientación a objetos bajo el lenguaje Java.

En el apartado de “Componentes y clases que intervienen” del Diseño técnico se presentará un diagrama base con las clases principales que intervendrán en cada capa dentro del patrón modelo vista controlador y el diseño en tres capas.

Tal y como se planteó en el apartado 3.3.1.7 Funcionalidades de reporte de datos, el componente de reporte tendrá dos tipos de modalidades para su ejecución. A partir de un proceso diario y la ejecución online.

El motor del proceso será el mismo con la distinción de la forma de emitir la petición y el tipo de reporte a generar.

Mientras la parte online permitía generar en tres frecuencias distintas (última hora, último turno y diario) en la modalidad asíncrona se estableció una simulación de envío de informe de diario.

El proceso por fases para generar el reporte será la siguiente:

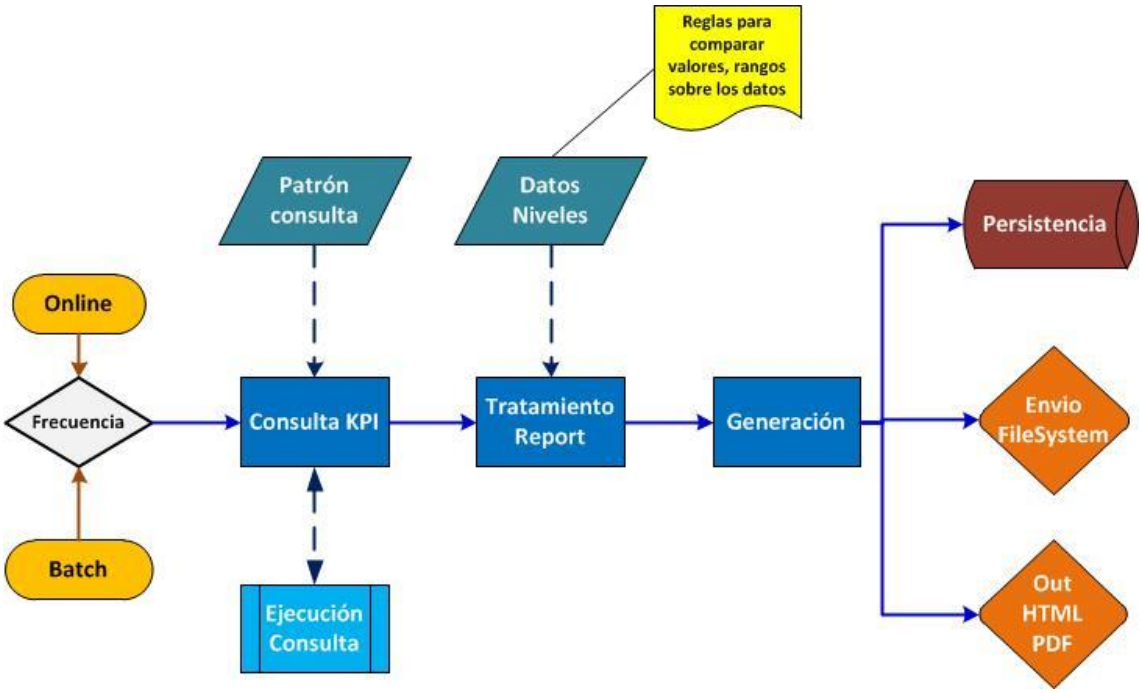


ILUSTRACIÓN 60: PROCESO DE CONSTRUCCIÓN DE REPORTE DE DATOS

Según se puede ver en el diagrama de flujo, el proceso de generación automática lo puede iniciar la funcionalidad online de Reporte o un proceso Batch.

Dependiendo de la frecuencia de la selección (el proceso batch será diario), el flujo se dirige a la fase de extracción de datos.

La Consulta de los indicadores clave de datos se basará en un patrón de consulta donde se le insertarán los datos que vienen del flujo, principalmente el tipo de informe (Diario, última hora o último turno) y la fecha para generar la consulta.

Una vez generada automáticamente el código de la consulta (basado en lenguaje SQL) se ejecutará el subproceso de ejecución de consulta donde devolverá la estructura de datos que será el input para al tratamiento del report.

El report contendrá la lógica para construir el reporte a partir del resultado de la consulta en forma de una estructura de datos y los datos de niveles de comparación.

Una vez construida la lógica del reporte se generará una estructura de salida para enviar los datos a un formato legible en html y/o pdf.

Para cerrar el informe se persistirá su evidencia en la Base de datos para que se puedan realizar consultas sobre ellos y se depositará en una ruta del filesystem del sistema.

Estructura base

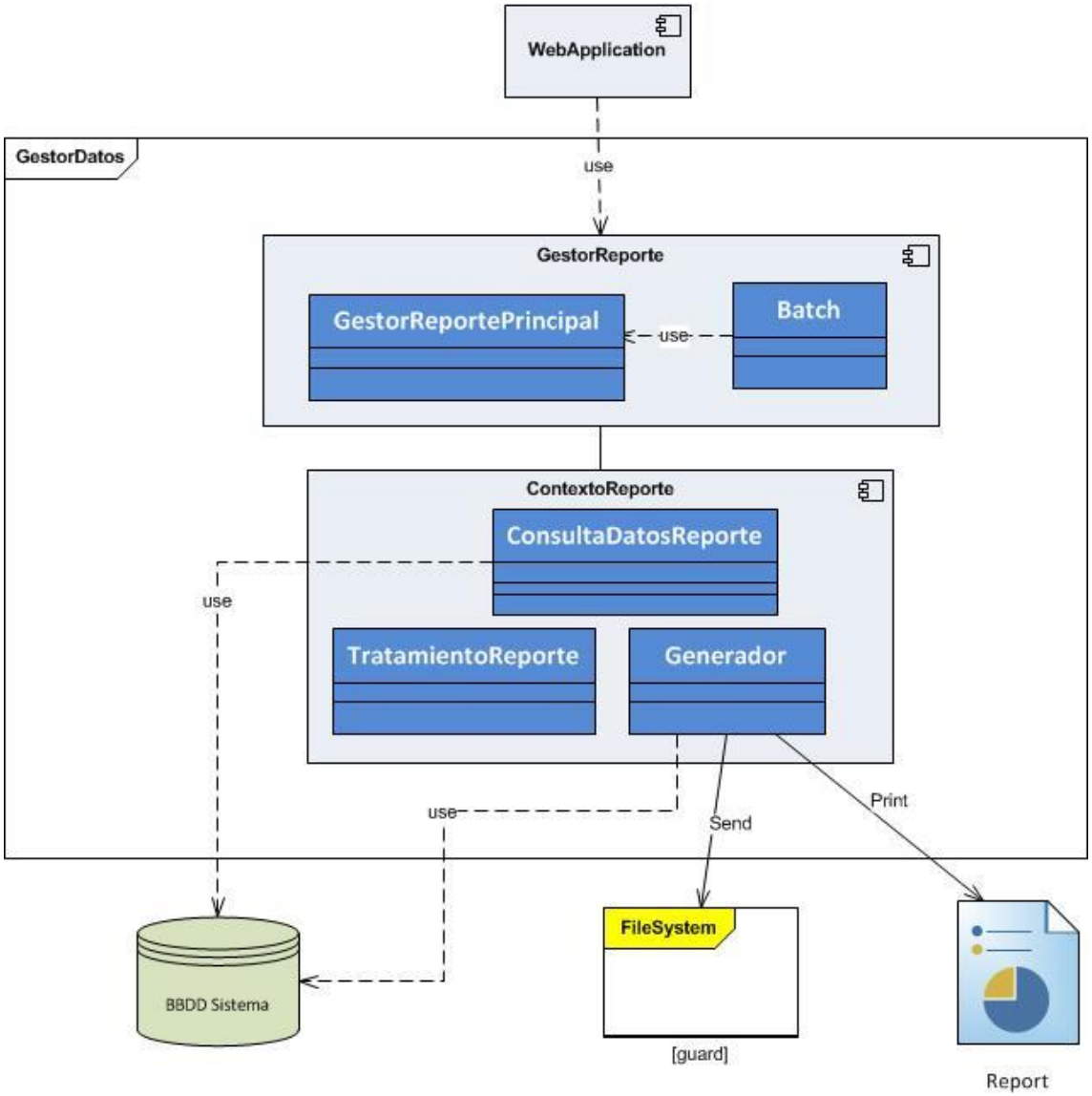


ILUSTRACIÓN 61: ESTRUCTURA DE COMPONENTES PARA GENERAR EL REPORTING

Para llevar a cabo las fases del proceso de reporte visto al inicio de este subapartado, se implementará la arquitectura base descrita en la imagen. Dentro de la arquitectura lógica del sistema, estará incluida en la capa de reporte donde se repartirá la responsabilidad en siguientes componentes:

- **GestorReporte:**
  - GestorReportePrincipal: Encapsulará la lógica para crear el reporte a partir de los parámetros recibidos del componente online de gestión de reporte y la del proceso Batch. Llama al

objeto extractor de datos para que ejecute la consulta preconstruida.

- Batch: Proceso asíncrono que invoca al gestorReportePrincipal para generar el report, envía la fecha para que ejecute el método de reporte diario.

- **ContextoReporte:**
  - ConsultaDatosReporte: Ejecuta la consulta SQL y encapsula los datos en objetos de las entidades relacionadas con el reporte.
  - TratamientoReporte: Compara los datos adquiridos de la consulta con las reglas establecidas en el apartado de simulaciones para informar el reporte con los indicadores comparados con las reglas para mostrar los porcentajes de impactos y recomendaciones ante problemas de cuellos de botella.
  - Generador: La instancia de este objeto ejecutará tres funciones:
    - Envió de reporte a filesystem
    - Persistencia del registro de reporte
    - Impresión de reporte (solo caso de uso online).

3.2.3. DISEÑO TÉCNICO

3.2.3.1. MODELO DE DATOS

El sistema contará con un modelo de datos basado en un sistema de persistencia relacional.

Para generar su diseño base se parte del modelo de dominio y el modelo global de componentes especificación funcional, para ello se identifican las diferentes entidades y su interrelación que intervendrán para persistir el sistema:

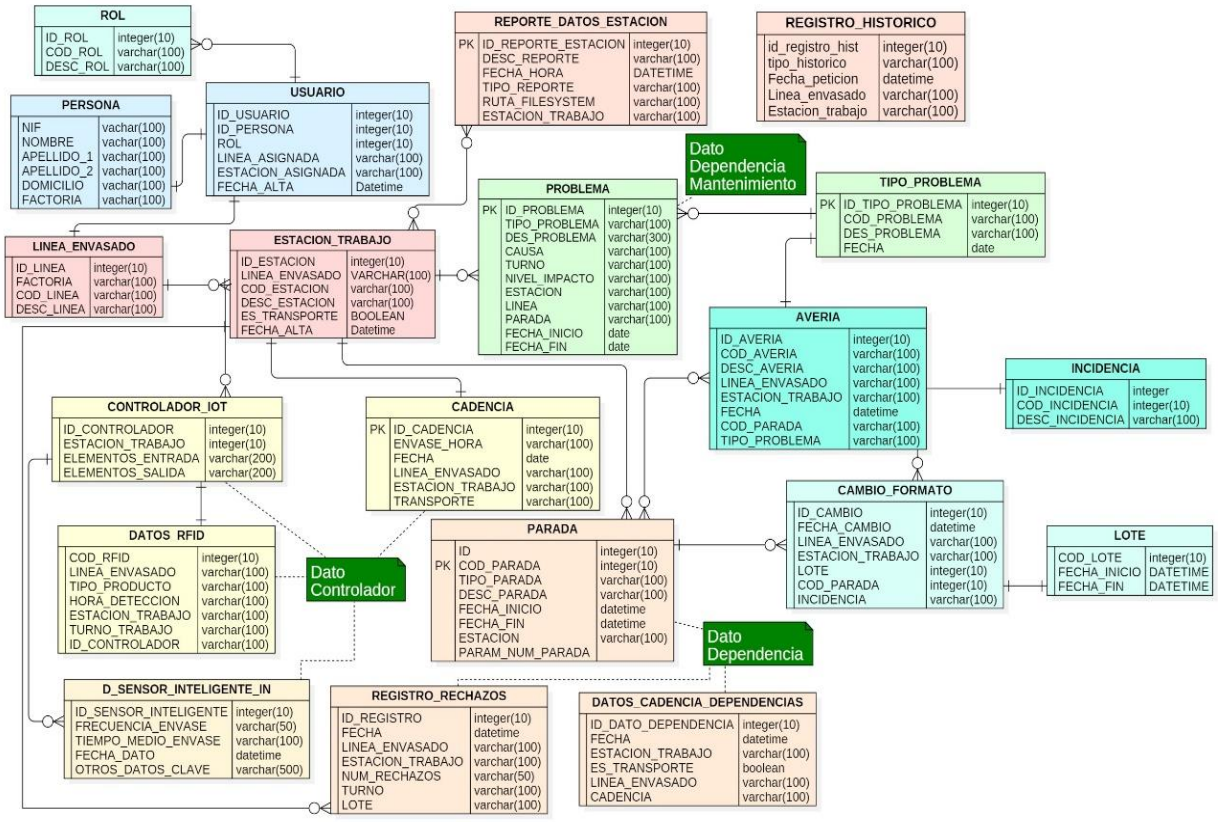


ILUSTRACIÓN 62: MODELO DE DATOS DEL SISTEMA

El detalle de cada tabla se describe en el Anexo 6: Detalle de tablas del modelo de datos.

3.2.3.2. COMPONENTES Y CLASES QUE INTERVIENEN

Tal y como se indicó en la arquitectura Web, se seguirá un diseño de tres capas para separar por responsabilidades a las diferentes capas que intervienen: Presentación, negocio y datos:

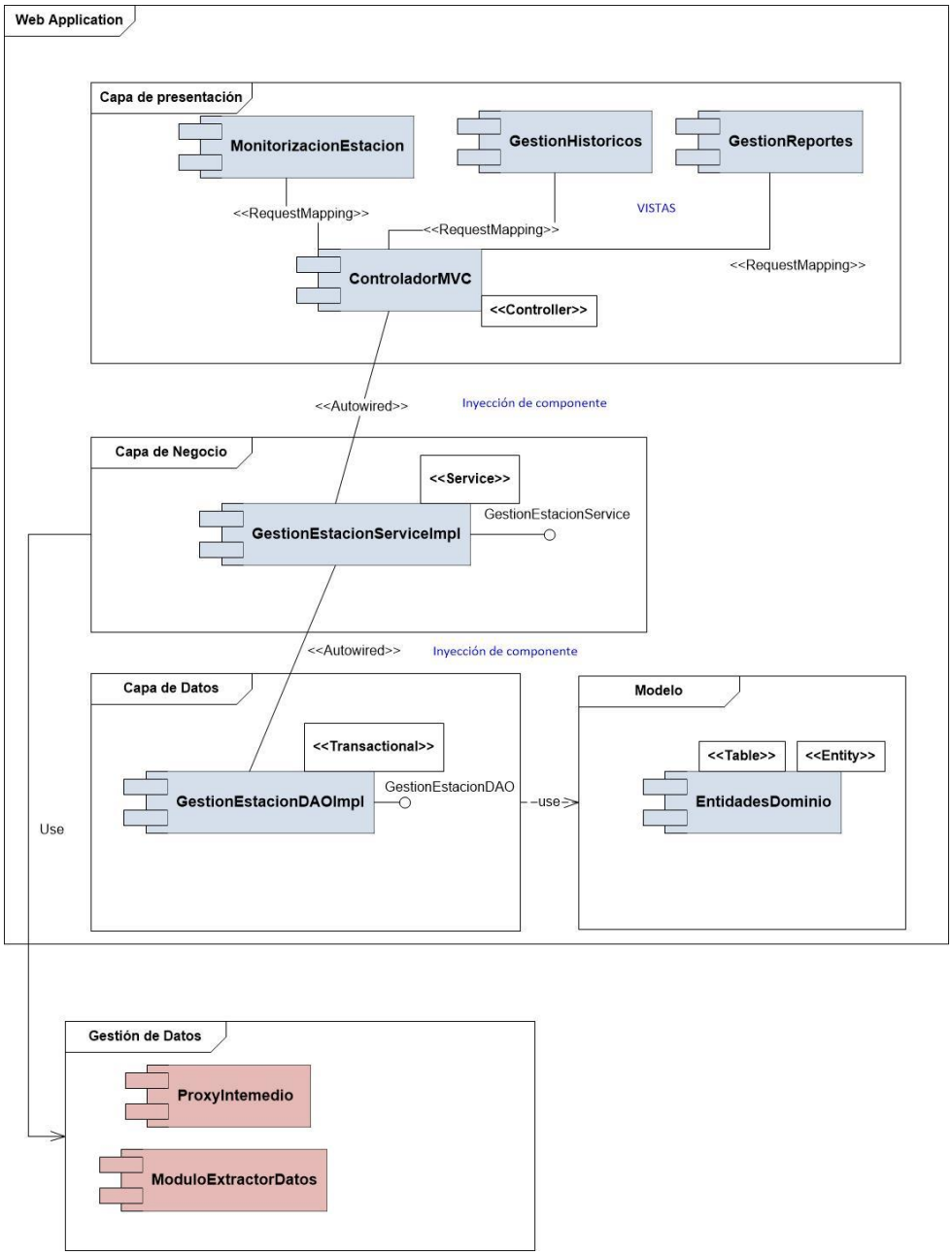


ILUSTRACIÓN 63: COMPONENTES PRINCIPALES DE ARQUITECTURA WEB DE 3 CAPAS

La tecnología que se utilizará para este diseño será bajo lenguaje JAVA J2EE y Spring Framework. Esto es porqué la pila de componentes de Spring cubre perfectamente un diseño de aplicación Web en tres capas bajo patrón MVC además de elementos transversales como la intercepción a través del módulo AOP y orientación a aspectos y la gestión transversal de datos a través del contenedor Central. Gráficamente se puede ver en el siguiente detalle del motor de Spring:

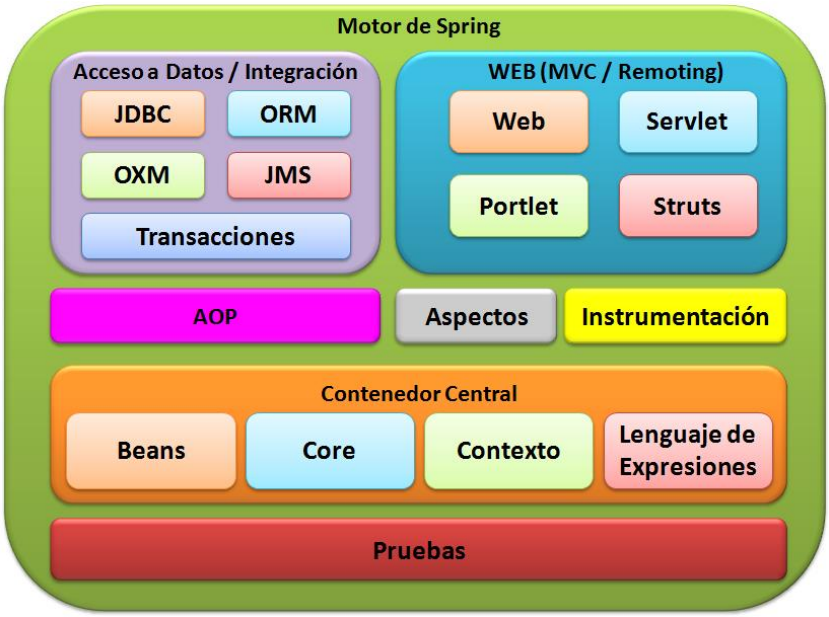


ILUSTRACIÓN 64: PILA DE MÓDULOS SPRING

Los diferentes módulos se pueden ver en detalle práctico en la siguiente url: https://spring.io/projects

A partir de aquí, el aplicativo Web, estará compuesto por tres capas principales:

- **Capa de Presentación** donde quedará implementado el patrón MVC menos el componente Modelo que estará a nivel de negocio y datos. Básicamente está formado por el controlador, las vistas y el motor de Spring MVC [SPR01] basado en anotaciones que resuelve e identifica a través de un DSL interno los componentes del patrón y realiza las inyecciones de componentes entre capas.

  Para el dominio del sistema, se definen tres interfaces de usuario principales: Monitorización, gestión de históricos y reportes y un componente controlador donde se orquestará el trasiego entre las capas inferiores (negocio y datos) y las interfaces de usuario.
- **Capa de negocio:** La capa de negocio implementará la clase de negocio GestionEstacionService para manejar la lógica de negocio en esta parte del sistema. Resolverá las peticiones que provendrán del controlador respecto a las funcionalidades de monitorización y la gestión para

mostrar los históricos y generar el reporte. En la implantación de la solución prototipo se mostrarán los detalles a nivel de código de estas funcionalidades.

- **Capa de datos:** Esta capa trabaja en el contexto de persistencia y transaccionalidad de datos. En el componente se modelarán las clases DAO (Data Acces Object) que resolverán las actuaciones contra la base de datos. La clase principal será GestionEstacionDAOImpl que implementará la interfaz GestionEstacionDAO.

En este sistema se utilizarán anotaciones como elemento cualificado para resolver el contexto de capas sobre el motor de Spring Core [SPR02].

El framework Spring permite gestionar sus artefactos y componentes a través de dos modalidades, a partir de un XML con resolución de nombres o bien a partir de anotaciones que identifican el tipo de objeto y que reglas ha de seguir en cuanto a su ejecución de código. Las dos opciones son recomendadas y se puede usar de forma híbrida. En este caso se utilizan anotaciones para no sobrecargar el prototipo utilizando ambas opciones.

Anotaciones principales que se utilizarán:

<<RequestMapping>>	Asigna la URL con la que mapeará un objeto una vez se haya ejecutado y el método http utilizado (GET o POST)
<<Controller>>	Especifica el tipo de clase Controlador que maneja al motor Spring dentro del patrón MVC.
<<Autowired>>	Inyecta un componente de otro contexto en una Clase para poder instanciarlo.
<<Transactional>>	Convierte a una clase o método en un objeto que permite un proceso transaccional contra una Base de Datos.
<<Repository>>	Asocia una clase a un acceso lógico a una BBDD, se asocian a las clases tipo DAO.
<<Entity>>	Identifica como entidad relacional a una clase
<<Table>>	Mapea una clase a una tabla que pertenece a un esquema de BBDD

Pila base de componentes tecnológicos que se usarán:

ILUSTRACIÓN 65: COMPONENTES TECNOLÓGICOS DEL DISEÑO WEB

JSP	Página de servidor Java. Estructura basada en librería de etiquetas core para desarrollar interfaz de usuario
Spring MVC	Módulo del framework Spring que Implementa la lógica del patrón modelo vista controlador.
JPA + Hibernate	Motor de persistencia Java y framework para gestionar la persistencia
MySQL	Motor de base de datos relacional Oracle MySQL.

3.2.3.3. DIAGRAMAS DE SECUENCIA

En el anexo 7 se muestra el detalle de los principales diagramas de primer nivel de secuencia con los pasos entre los componentes y clases que se implementarán para automatizar las funcionalidades diseñadas para el presente sistema:

- Monitorización Estación.
- Gestión históricos.

- Reporte online.

En cuanto al proceso generado por la detección de los sensores del controlador IoT, se muestran los siguientes diagramas de secuencia en el diseño para el procesado de la señal del sensor hasta su persistencia:

- Sensor RFID.
- Sensor cadencia.
- Sensor entrada.

Por último, en el mismo anexo, se muestra un diagrama de secuencia a alto nivel con una iniciativa de modelo de secuencia de envío a partir del protocolo TCP del controlador externo a las dependencias del presente sistema.

3.3. IMPLEMENTACIÓN DE LA SOLUCIÓN

Para el alcance del presente trabajo se ha preparado un prototipo que mostrará el resultado de las maquetas vistas en el apartado 3.1.1.8 “Pantallas maqueta del prototipo” de este documento.

Para ello se ha creado e implementado una estructura de código mínimo que muestra cómo serían la interface de usuario en una monitorización de datos, resultado de un reporte diario y un gráfico de la funcionalidad de gestión de históricos sobre el sistema solución diseñado en el trabajo.

Dado que este sistema precisará de datos en tiempo real, estos se simulan implementando una función de test.

En las siguientes líneas, se hace una descripción de los componentes que se han optado por utilizar como base para la implementación del sistema.

3.3.1. PILA DE COMPONENTES INTEGRADOS

ILUSTRACIÓN 66: PILA DE COMPONENTES INTEGRADOS EN IMPLEMENTACIÓN

3.3.1.1. COMPONENTES INTEGRADOS EN EL PROTOTIPO

El prototipo está compuesto por cuatro componentes principales:

- **Lado Cliente**: Compuesto principalmente por la librería de código abierto Bootstrap [BOO01] basada en la pila de html5, css3 y javascript. Concretamente se utiliza la opción de código abierto Starmin. El mecanismo que se utiliza para implementar estas librerías son la adaptación de los tags html en páginas de servidor Java (JSP) para que permita un ciclo completo de peticiones cliente-servidor con la ayuda del patrón modelo vista controlador.

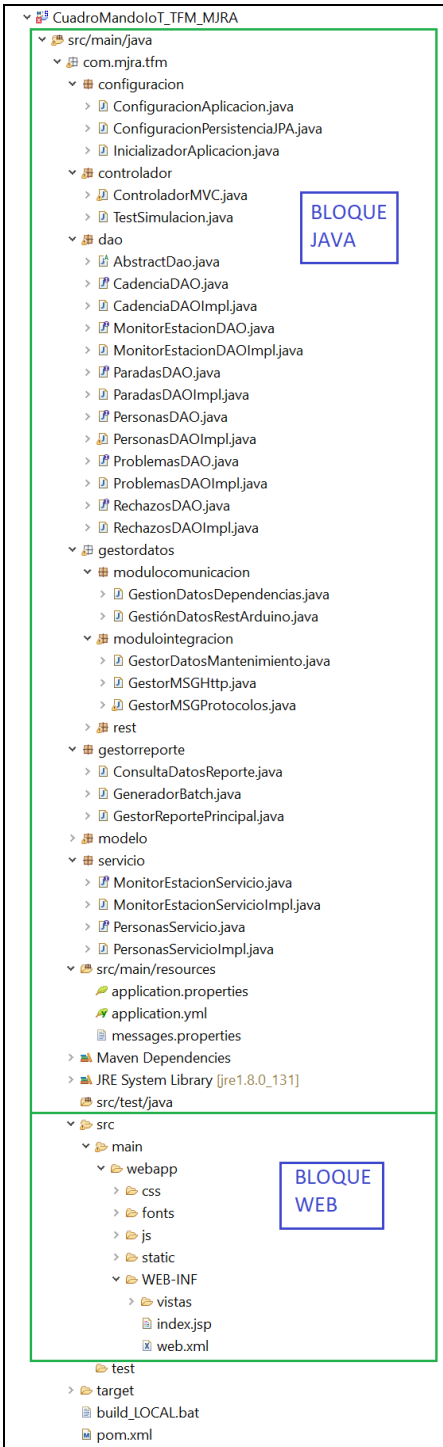
  Una vez estas páginas han sido publicadas desde la parte servidor, el navegador cliente mostrará todo el código html traducido y las variables

Java enviadas a través del protocolo http por parte del dispensador de Servlet del framework Sprint Web.

- **Lado Servidor**: Integrado con los componentes vistos en el apartado 3.2.3.2 de este documento. Se basa principalmente en una arquitectura de tres capas Presentación, negocio y datos. Para el prototipo se prepara la estructura del código, pero se representan datos emulados estáticos para mostrar parte de las funcionalidades de monitorización, gestión de históricos y reporte diario. El Servidor elegido es un contenedor Tomcat [TOM01] versión 9, porqué es un contenedor ligero basado en la unidad Java Servlet y este integra perfectamente con cualquier módulo de framework Spring escogido.
- **Proxy datos - Microservicios**: Componente independiente que presta servicio a la aplicación principal de monitorización, implementado a través del framework Spring boot [SPR03]. Trabaja como un servicio Intermedio para consumir datos de dependencias, aplicativo de mantenimiento externos, los datos del propio controlador Arduino y como dispensador de datos para que sean consumidos por otros sistemas. Técnicamente trabaja como una aplicación standalone con un servidor Tomcat empotrado. Este permite publicar y consumir servicios en la Web.
- **Microcontrolador**: Adquiere los datos de entradas y salidas del controlador Arduino y los prepara para ser enviados a través del Servidor Web empotrado en el hardware Arduino Yun.
- **Servidor Web Arduino**: El microprocesador de Arduino Yun contiene el sistema lilino basado en Linux, este a su vez, contiene un servidor Web y una API REST para poder emitir los datos del microcontrolador. La aplicación prototipo Arduino compilada en C++ contiene la emulación para la publicación de datos de los sensores a través de este servidor Web.

3.3.2. ESTRUCTURA DEL CÓDIGO IMPLEMENTADO

3.3.2.1. SISTEMA WEB PRINCIPAL



El código está estructurado en un proyecto de integración continua Maven. Entre otros aspectos, esta estructuración permite la escalabilidad y cohesión de componentes del proyecto de forma automatizada a través de la gestión de un solo descriptor XML.

El proyecto Maven divide la parte Java y la parte Web que será publicada en cliente.

El bloque Java contiene la definición y parte de implementación de los componentes técnicos vistos a lo largo del capítulo 3:

- MVC:
  - Controlador
  - Servicio (Negocio)
  - DAO
  - Modelo
- GestorDatos:
  - Modulo Comunicación
  - Modulo Integración
- GestorReporte

El bloque Web contiene todo el código html, css, javascript y componentes del Document Object Model necesarios para publicar la aplicación Web.

ILUSTRACIÓN 67: ESTRUCTURA DE CÓDIGO SISTEMA WEB

El prototipo se centra en implementar parte de las funcionalidades de monitorización de estación, gestión de históricos y gestión de reporte.

Para ello, se emulan los datos a partir de una clase llamada TestSimulacion. En este se simulan los datos que deberían de ser adquiridos consultando al modelo de datos o bien datos adquiridos del microcontrolador. En el caso de este recorte se están simulando los datos del gráfico de cadencias:

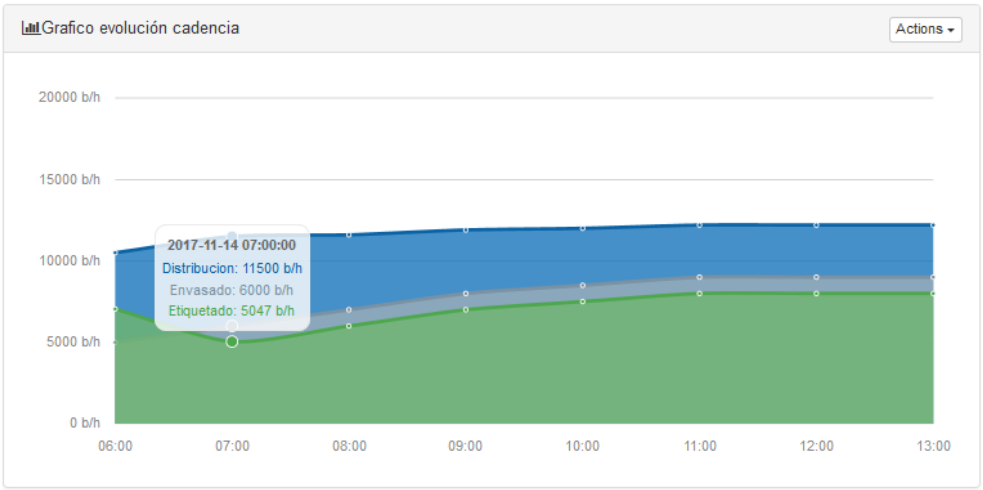


ILUSTRACIÓN 68: GRÁFICO SIMULACIÓN EVOLUCIÓN CADENCIA ESTACIONES DE DISTRIBUCIÓN, ENVASADO Y ETIQUETADO.

```java
/**
 * Simulacion del grafico area cadencias
 * Método simulación TEST 14-11-2017 Datos cadencias turno Mañana
 * @param indice
 * @return
 */
public GraficoMonitorArea datosAuxiliarArea(int indice){

    GraficoMonitorArea graficoMonitorArea = new GraficoMonitorArea();

    switch (indice) {
    case 0:

        graficoMonitorArea.setDistribucion("10500");
        graficoMonitorArea.setEnvasado("5000");
        graficoMonitorArea.setEtiquetado("7047");
        graficoMonitorArea.setPeriodo("2017-11-14 06:00:00");

        break;
    case 1:

        graficoMonitorArea.setDistribucion("11500");
        graficoMonitorArea.setEnvasado("6000");
        graficoMonitorArea.setEtiquetado("5047");
        graficoMonitorArea.setPeriodo("2017-11-14 07:00:00");

        break;
    case 2:

        graficoMonitorArea.setDistribucion("11600");
        graficoMonitorArea.setEnvasado("7000");
        graficoMonitorArea.setEtiquetado("6000");
        graficoMonitorArea.setPeriodo("2017-11-14 08:00:00");

        break;
    case 3:

        graficoMonitorArea.setDistribucion("11900");
        graficoMonitorArea.setEnvasado("8000");
        graficoMonitorArea.setEtiquetado("7000");
        graficoMonitorArea.setPeriodo("2017-11-14 09:00:00");

        break;
```

ILUSTRACIÓN 69: SIMULACIÓN DE CASOS DE MONITORIZACIÓN

Se crean las entidades que serán mapeadas en el modelo de datos, tal y como se muestra en el siguiente ejemplo, a la izquierda se puede ver la lista de clases pertenecientes a las entidades del modelo de datos, a la derecha un recorte de la entidad Cadencia.

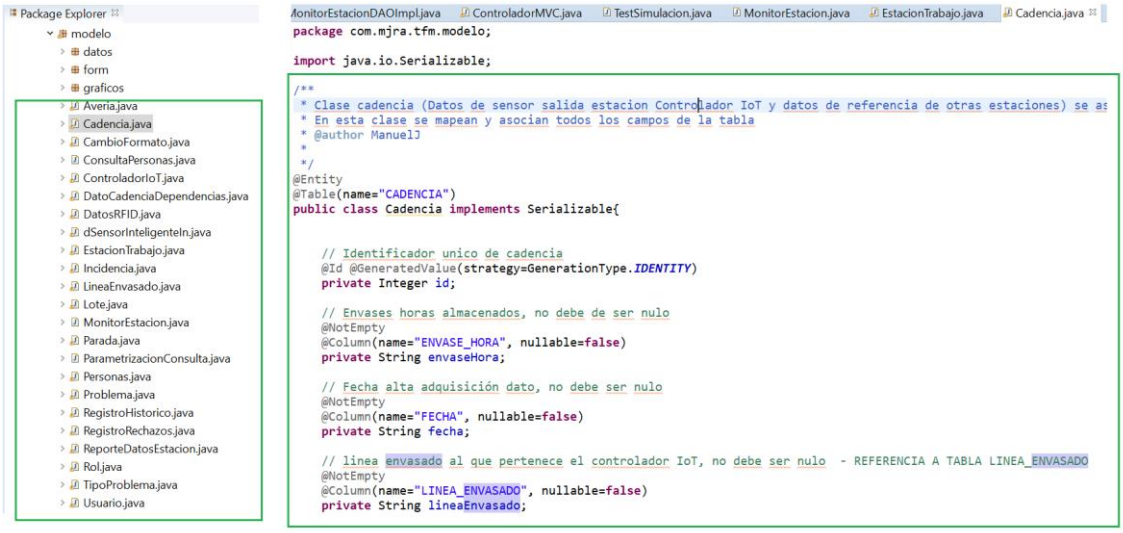


ILUSTRACIÓN 70: ESTRUCTURA DE ENTIDADES DEL MODELO DE DATOS

Para ver con mayor detalle tanto las clases del Modelo como clase Test de simulación se puede revisar en el código fuente implementado.

Para el prototipo se implementan métodos para poder mostrar una monitorización de estación, gráfico histórico cadencia estación y reporte diario, en el siguiente recorte se muestra la declaración del método:

```java
    /**
     * TFM
     * Método principal para mostrar la monitorización del sistema
     */
    public String getMonitorEstacion(ModelMap model) {


    /**
     * TFM
     * Acción genera la página principal de gestión de históricos
     */
    public String getGestionHistorico(ModelMap model) {


/**
 * TFM Consulta histórico, una vez seleccionado un histórico por fecha, se crea una petición para mostrar el gráfico demandado
 * @param consultaHistoricos
 * @param result
 * @param model
 * @return
 */
    public String consultaHistoricoPOST(@Valid ConsultaHistoricos consultaHistoricos, BindingResult result, ModelMap model) {

    /**
     * TFM
     * Método principal para la gestión de reporte
     */
    public String getGestionReporte(ModelMap model) {



    /**
     * Gestion Reporte, Método para generar y consultar reportes según la opción escogida en la UI
     * @param consultaHistoricos
     * @param result
     * @param model
     * @return
     */
        public String gestionReportePOST(@Valid GestionReporteDatos gestionReporteDatos, BindingResult result, ModelMap model) {

    /**
     * TFM
     * Método Prueba grafico
     */
    public String getGraficoCadenciaEstacion(ModelMap model) {

    /**
     * TFM
     * Método Prueba grafico
     */
    public String getReporteOnline(ModelMap model) {
```

ILUSTRACIÓN 71: RECORTE CLASE CONTROLADORMVC, MÉTODOS GEST Y POST FUNCIONALIDADES IMPLEMENTADAS

El siguiente recorte representa la implementación específica del método getMonitorEstacion, según se puede ver en el recorte, los datos para el prototipo se adquieren de la clase TestSimulacion:

```java
/**
 * TFM
 * Método principal para mostrar la monitorización del sistema
 */
@RequestMapping(value = { "/MonitorEstacion" }, method = RequestMethod.GET)
public String getMonitorEstacion(ModelMap model) {


    /****************** TEST ***************************/
        MonitorEstacion monitorEstacion = new MonitorEstacion();
        // TEST datos cabecera
        monitorEstacion.setCadenciaEstacion("5000");
        monitorEstacion.setCadenciaTransporte("8901");
        monitorEstacion.setNumParadasTurno(12);
        monitorEstacion.setNumProblemasEstacion(14);
        TestSimulacion testSimulacion = new TestSimulacion();
        /** TEST datos área */
        List listaGraficoCadenciaArea = new ArrayList<GraficoMonitorArea>(8);
        for (int i = 0; i < 8; i++) {
            // nuevo objeto
            GraficoMonitorArea graficoMonitorArea = testSimulacion.datosAuxiliarArea(i);
            listaGraficoCadenciaArea.add(i, graficoMonitorArea);
        }
        monitorEstacion.setListaGraficoCadenciaArea(listaGraficoCadenciaArea);
        /** TEST datos diagrama barras rechazos (listado de semana) */
        List listaGraficoRegistroRechazos = new ArrayList<RegistroRechazos>(7);
        for (int i = 0; i < 8; i++) {
            // nuevo objeto
            RegistroRechazos registroRechazos = testSimulacion.datosAuxiliarBarrasRechazos(i);
            listaGraficoRegistroRechazos.add(i, registroRechazos);
        }
        monitorEstacion.setListaRechazosEstaciones(listaGraficoRegistroRechazos);
        /** TEST datos diagrama donut rechazos 3 turnos */
        List listaGraficoDonutRechazos = new ArrayList<RegistroRechazos>(8);
        for (int i = 0; i < 8; i++) {
            // nuevo objeto
            RegistroRechazos registroRechazos = testSimulacion.datosAuxiliarDonutRechazos(i);
            listaGraficoDonutRechazos.add(i, registroRechazos);
        }
        monitorEstacion.setListaNumRechazosJornada(listaGraficoDonutRechazos);
        /** TEST datos paradas, estos datos habrán que calcularlos a partir de la consulta de paradas */
        List listaGraficoRegistroParadas = new ArrayList<GraficoRegistro>(8);
        for (int i = 0; i < 8; i++) {
            // nuevo objeto
            GraficoRegistro graficoRegistroParada = testSimulacion.datosAuxiliarRegistroParadas(i);
            listaGraficoRegistroParadas.add(i,graficoRegistroParada);
        }
        monitorEstacion.setListaNumParadas(listaGraficoRegistroParadas);
        /** TEST datos paradas, estos datos habrán que calcularlos a partir de la consulta de paradas, en este caso numero paradas tres turnos */
        List listaGraficoDonutParadas = new ArrayList<GraficoRegistro>(3);
        for (int i = 0; i < 3; i++) {
            // nuevo objeto
            GraficoRegistro graficoRegistroParadaTurno = testSimulacion.datosAuxiliarDonutParadas(i);
            listaGraficoDonutParadas.add(i, graficoRegistroParadaTurno);
        }
        monitorEstacion.setListaNumParadasJornada(listaGraficoDonutParadas);
        /** TEST datos problemas, listado de número de problemas 7 dias, */
        List listaGraficoRegistroProblemas = new ArrayList<GraficoRegistro>(7);
        for (int i = 0; i < 7; i++) {
            // nuevo objeto
            GraficoRegistro graficoRegistroProblemas = testSimulacion.datosAuxiliarRegistroProblemas(i);
            listaGraficoRegistroProblemas.add(i,graficoRegistroProblemas);
        }
        monitorEstacion.setListaNumProblemas(listaGraficoRegistroProblemas);
        /** TEST datos diagrama donut problemas 3 turnos */
        List listaGraficoDonutProblemas = new ArrayList<GraficoRegistro>(8);
        for (int i = 0; i < 3; i++) {
            // nuevo objeto
            GraficoRegistro graficoRegistroProblemasTurno = testSimulacion.datosAuxiliarDonutProblemas(i);
            listaGraficoDonutProblemas.add(i, graficoRegistroProblemasTurno);
        }
        monitorEstacion.setListaNumProblemasJornada(listaGraficoDonutProblemas);


        model.addAttribute("monitorEstacion", monitorEstacion);


      return "MonitorEstacion";
}
```

ILUSTRACIÓN 72: RECORTE CLASE TEST SIMULACIÓN

Estos datos serán enviados a la parte Web, donde están representadas las distintas vistas, en el siguiente recorte se muestran las vistas JSP del prototipo:

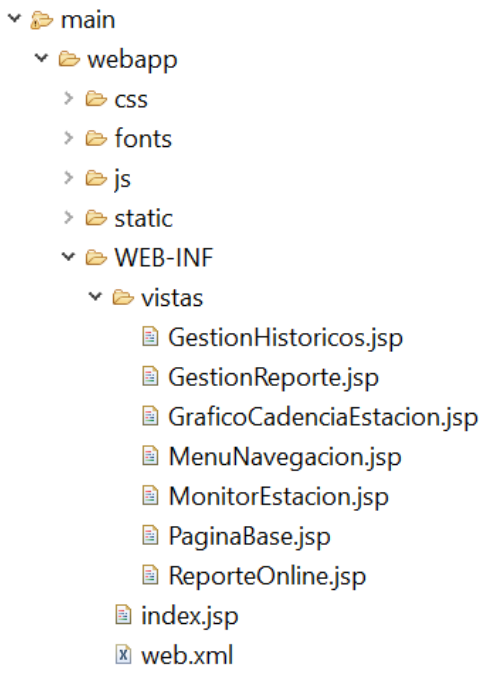


ILUSTRACIÓN 73: ESTRUCTURA WEB

La página de servidor Java adaptada a Bootstrap contendrá la siguiente estructura:

ILUSTRACIÓN 74: ARQUITECTURA PÁGINA BOOTSTRAP HTML ADAPTADA A JSP

Dada la extensión del contenido de las páginas, estas se podrán evaluar examinando el código fuente.

Esta estructura permitirá visualizar un diseño como el del siguiente recorte:

ILUSTRACIÓN 75: PÁGINA PRINCIPAL MONITORIZACIÓN ESTACIÓN

La estructura de vistas resultado se detalla en el capítulo 4 de evaluación.

3.3.2.2. MÓDULO PROXY INTEGRACIÓN DE DATOS Y ARDUINO

Se crea dentro del prototipo el módulo proxy, según el diseño del sistema, este representaba inicialmente la integración con datos de dependencias externas. Dentro de la implementación se ve la posibilidad también para el uso de integración de datos del controlador IoT, además de ser un sistema para crear una API con servicios REST como utilización global.

La estructura quedará de la siguiente forma:

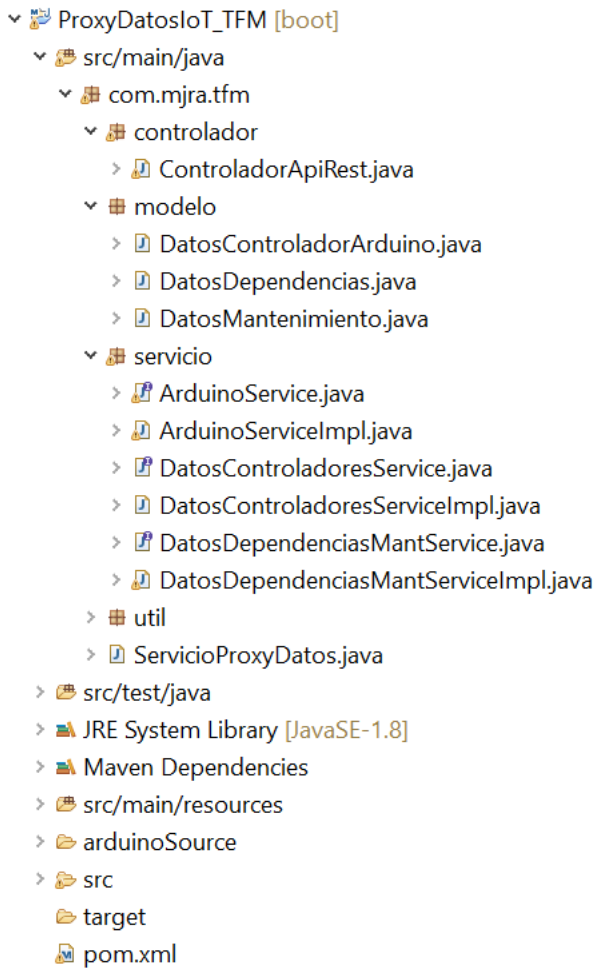


ILUSTRACIÓN 76: ESTRUCTURA CARPETAS CÓDIGO PROXY

Según se puede ver, la estructura consta de tres componentes principales:

Controlador: Motor del microservicio, se basa en implementar funciones a partir de los métodos GET y POST, para recoger y publicar datos de los servicios.

Servicios: Se basa en tres métodos principales, datos adquiridos del controlador Arduino, dependencias externas y datos de un hipotético aplicativo de mantenimiento.

Modelo: Datos susceptibles a persistirlos en el modelo de datos del sistema.

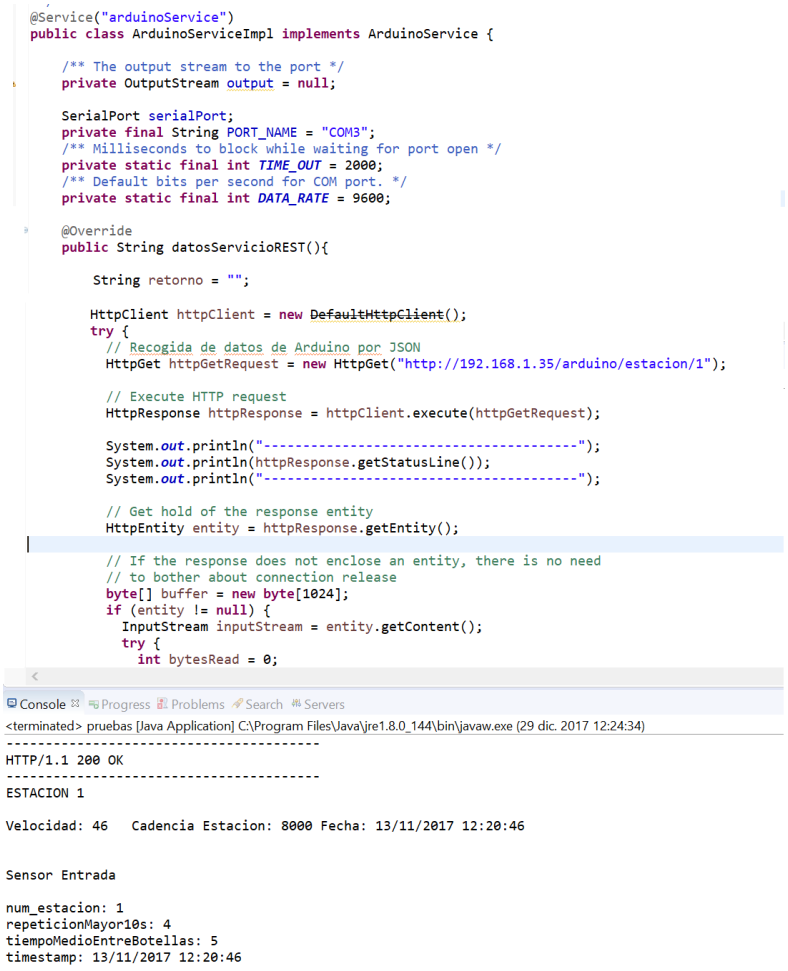
Para el prototipo, se implementa un test para verificar la conexión con el sistema del controlador Arduino YUN y adquirir los datos emulados.

Para ello, la clase ControladorApiRest contendrá el método getDatosSensor

```java
/**
 * Clase Proxy y Microservicios  para el servicio Rest. Intercepta datos dependencias, IoT y Mantenimiento. Puede publicar microservicios
 * para otras estaciones
 * los métodos GET, PUT Y POST para poder realizar el tratamiento de datos en la API REST que se genere
 * El motor está basado en spring boot que publicará los servicios, está basado en módulos Spring MVC, web y AOP 4.3.5
 * @author ManuelJ
 *
 */
@RestController
@RequestMapping("/api")
public class ControladorApiRest {

    public static final Logger logger = LoggerFactory.getLogger(ControladorApiRest.class);

    @Autowired
    ArduinoService arduinoService;
    DatosDependenciasMantService datosMantenimientoService;


    /**
     * Método GET para capturar los datos de Arduino y publicarlos en el servicio
     * Estos podrán ser adquiridos desde cualquier cliente que adapte esta API
     * @return
     */
    @RequestMapping(value = "/datosArduino/", method = RequestMethod.GET)
    public ResponseEntity<String> getDatosSensor() {
        String resultado = arduinoService.datosServicioREST();
        if (resultado== null) {
            return new ResponseEntity(HttpStatus.NO_CONTENT);
        }
        System.out.println("ENTRADO");
        return new ResponseEntity<String>(resultado, HttpStatus.OK);
    }
```

ILUSTRACIÓN 77: RECORTE CLASE CONTROLADORAPIREST

Esta a su vez llamará al cliente REST del Servicio Web emitido por el controlador Arduino YUN:



```
@Service("arduinoService")
public class ArduinoServiceImpl implements ArduinoService {

    /** The output stream to the port */
    private OutputStream output = null;

    SerialPort serialPort;
    private final String PORT_NAME = "COM3";
    /** Milliseconds to block while waiting for port open */
    private static final int TIME_OUT = 2000;
    /** Default bits per second for COM port. */
    private static final int DATA_RATE = 9600;

    @Override
    public String datosServicioREST(){

        String retorno = "";

        HttpClient httpClient = new DefaultHttpClient();
        try {
          // Recogida de datos de Arduino por JSON
          HttpGet httpGetRequest = new HttpGet("http://192.168.1.35/arduino/estacion/1");

          // Execute HTTP request
          HttpResponse httpResponse = httpClient.execute(httpGetRequest);

          System.out.println("----------------------------------------");
          System.out.println(httpResponse.getStatusLine());
          System.out.println("----------------------------------------");

          // Get hold of the response entity
          HttpEntity entity = httpResponse.getEntity();

          // If the response does not enclose an entity, there is no need
          // to bother about connection release
          byte[] buffer = new byte[1024];
          if (entity != null) {
            InputStream inputStream = entity.getContent();
            try {
              int bytesRead = 0;
```

```
Console  Progress  Problems  Search  Servers
<terminated> pruebas [Java Application] C:\Program Files\Java\jre1.8.0_144\bin\javaw.exe (29 dic. 2017 12:24:34)
----------------------------------------
HTTP/1.1 200 OK
----------------------------------------
ESTACION 1

Velocidad: 46   Cadencia Estacion: 8000 Fecha: 13/11/2017 12:20:46


Sensor Entrada

num_estacion: 1
repeticionMayor10s: 4
tiempoMedioEntreBotellas: 5
timestamp: 13/11/2017 12:20:46
```

ILUSTRACIÓN 78: RECORTE CLASE ADQUISICIÓN DATOS SERVICIO ARDUINO

Previamente, se crea una aplicación en Arduino de simulación a partir de las librerías puente que envía-consume datos de la periferia IO del controlador. Para probar el módulo proxy, se imprimen datos simulados que deberían de provenir de los sensores en la línea de envasado:

ILUSTRACIÓN 79: RECORTE CÓDIGO APLICACIÓN ARDUINO

Una vez ejecutado el programa, se publicará el servicio REST en el servidor Web de Arduino para que pueda ser accedido desde cualquier host que tenga permisos del Gateway donde esté conectado el dispositivo microcontrolador.

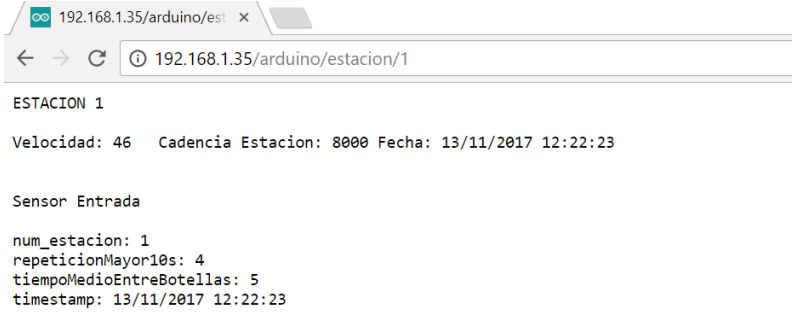


ILUSTRACIÓN 80: PUBLICACIÓN SERVICIO REST DE DATOS ARDUINO

En definitiva, el objetivo que persigue este módulo es la interconexión con otros sistemas con los que será necesaria interceptar información para que lo pueda consumir el sistema web principal. Las clases datos controladores manejarán la conexión / drivers contra los controladores principales de la estación para adquirir datos y otros aplicativos externos de los cuales también sean necesarios incorporar datos como el aplicativo con los datos de mantenimiento realizados en la estación monitorizada sobre la línea de producción.

3.3.2.3. RELACIÓN CON LOS COMPONENTES TÉCNICOS Y ARQUITECTÓNICOS

Tal y como se ha podido ver en los detalles de la implementación, se ha generado una estructura de código inicial siguiendo los pasos de la arquitectura y el diseño técnico definidos en el apartado de especificación técnica.

Se ha seguido la lógica definida por los patrones de diseño seguidos tal como el MVC o los que vienen implícitos en el propio framework de Spring tal como el facade y singleton.

No obstante, indicar que un desarrollo completo de todas las funcionalidades supondría evaluar completamente la efectividad del código y su puesta en escena en un escenario en tiempo real, teniendo en cuenta entro otros aspectos las comunicaciones, seguridad, integridad de datos, retardos en la transaccionalidad, etc.

En este sentido, habría que evaluar la cohesión entre todos los componentes software.

4. EVALUACIÓN

4.1. RESULTADOS OBTENIDOS

Una vez vista la estructura utilizada para la implementación, al ejecutar el sistema Web principal, se obtendrán los siguientes resultados:

Una vez puesto en marcha el Servidor de aplicaciones Tomcat sobre el host de la estación de trabajo, se autenticará un usuario para acceder a la aplicación:

ILUSTRACIÓN 81: PÁGINA AUTENTICACIÓN SISTEMA WEB

La interfaz de usuario para bootstrap estará implementado en modo “responsive”, esto quiere decir que el tamaño se adaptará a la pantalla desde donde se está visualizando.

Una vez autenticado, el objetivo será tener a primera vista los datos de la estación de trabajo monitorizada que muestren indicios de cuello de botella y gestionar las diferentes funcionalidades del sistema.

4.1.1. MONITORIZACIÓN DE ESTACIÓN DE ENVASADO

Con la presente vista se ha conseguido tener una visión global de las variables especificadas en la iniciativa de diseño y de este modo poder anticiparse a una problemática de tipo de cuello de botella.

ILUSTRACIÓN 82: PANTALLA MONITORIZACIÓN ESTACIÓN ENVASADO

El recorte representa la vista del cuadro de mando de la estación de trabajo. En un golpe de vista se verán los siguientes detalles.

- **Detalle Menú lateral y encabezado del cuadro de mando**

  Al desplegar los ítems del menú lateral y encabezado se encontrarán los siguientes datos:

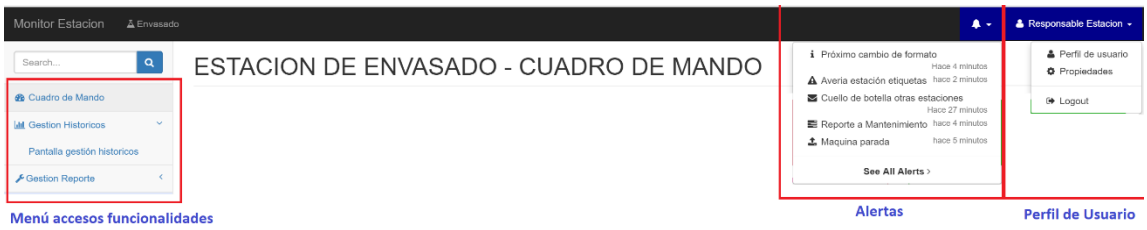


ILUSTRACIÓN 83: DETALLE CABECEERA Y MENÚ SISTEMA WEB

- **Detalle encabezado con los datos importantes para detectar cuello de botella:**
  - Cadencia de máquina respecto a su objetivo.
  - Cadencia de transporte de entrada
  - Problemas de envasado en turno
  - Paradas en turno

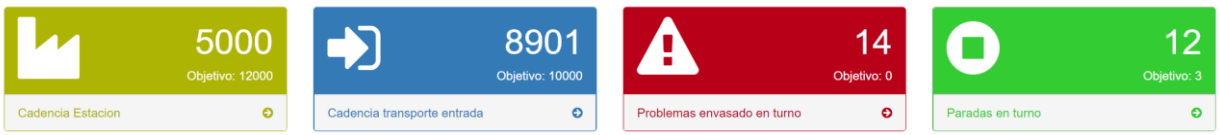


ILUSTRACIÓN 84: DETALLE CABECERA CON DATOS CLAVE ESTACIÓN

- **Detalle de gráfico evolución de cadencia:**

En la parte inferior se verá un gráfico con la evolución de cadencia de estación en las seis últimas horas de la estación de trabajo y sus dos estaciones más próximas. En la monitorización por pantalla si se sitúa el ratón o cursor encima de una hora del gráfico indicará sus valores instantáneos:

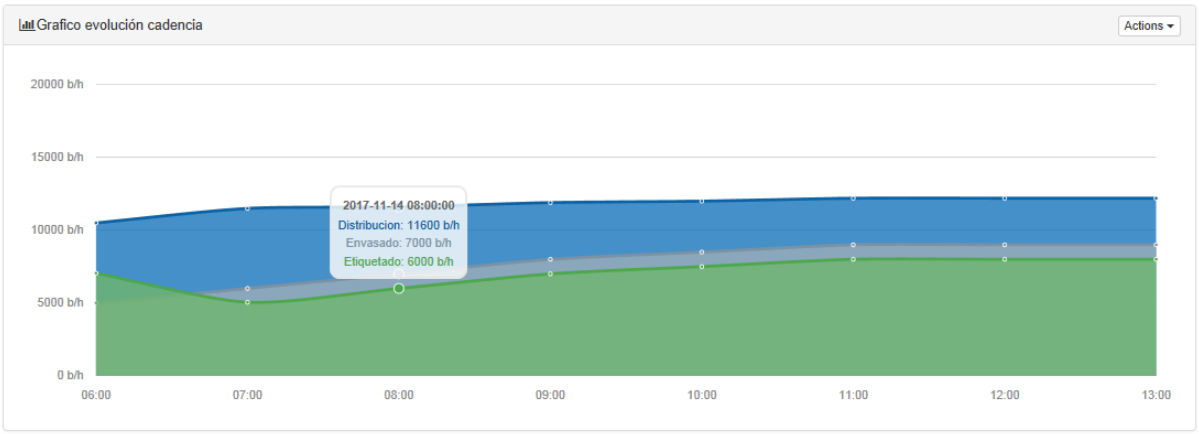


ILUSTRACIÓN 85: DETALLE GRÁFICO EVOLUCIÓN CADENCIA ESTACIÓN ENVASADO Y ESTACIONES DISTRIBUCIÓN Y ETIQUETADO

- **Detalle del panel con últimas notificaciones**

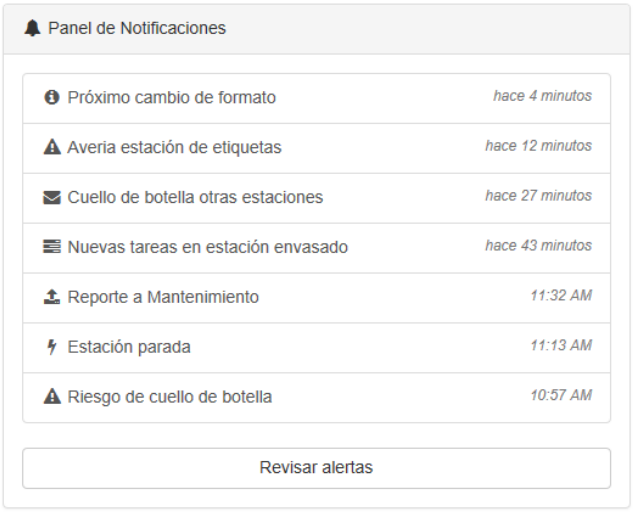


ILUSTRACIÓN 86: DETALLE PANEL NOTIFICACIONES

- **Detalle estatus de rechazos en turno y evolución**

En la parte izquierda se mostrará una tabla con los rechazos en turno de todas las estaciones de trabajo y a la derecha un gráfico de evolución que contrasta los rechazos de la semana con el objetivo a alcanzar.

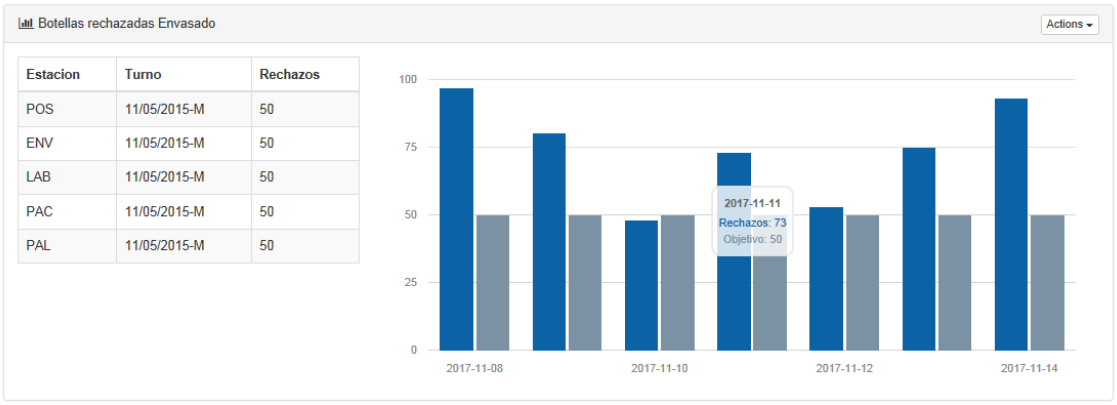


ILUSTRACIÓN 87: DETALLE GRÁFICO EVOLUCIÓN RECHAZOS

- **Detalle gráfico tipo Rosca**

En el gráfico tipo rosca muestra los rechazos en tres últimos turnos de trabajo. Tecnológicamente, el sistema gráfico utilizado es dinámico, según se sitúe el ratón encima de alguno de las tres áreas, mostrará en la parte central su valor.

ILUSTRACIÓN 88: DETALLE VISUALIZACIÓN DE DATOS DEL GRÁFICO DE ROSCA

- **Detalle listado y evolución de paradas y problemas en estación**

La monitorización cuenta además con dos gráficos más, siguiendo el mismo estilo que los rechazos: Paradas y problemas en estación.

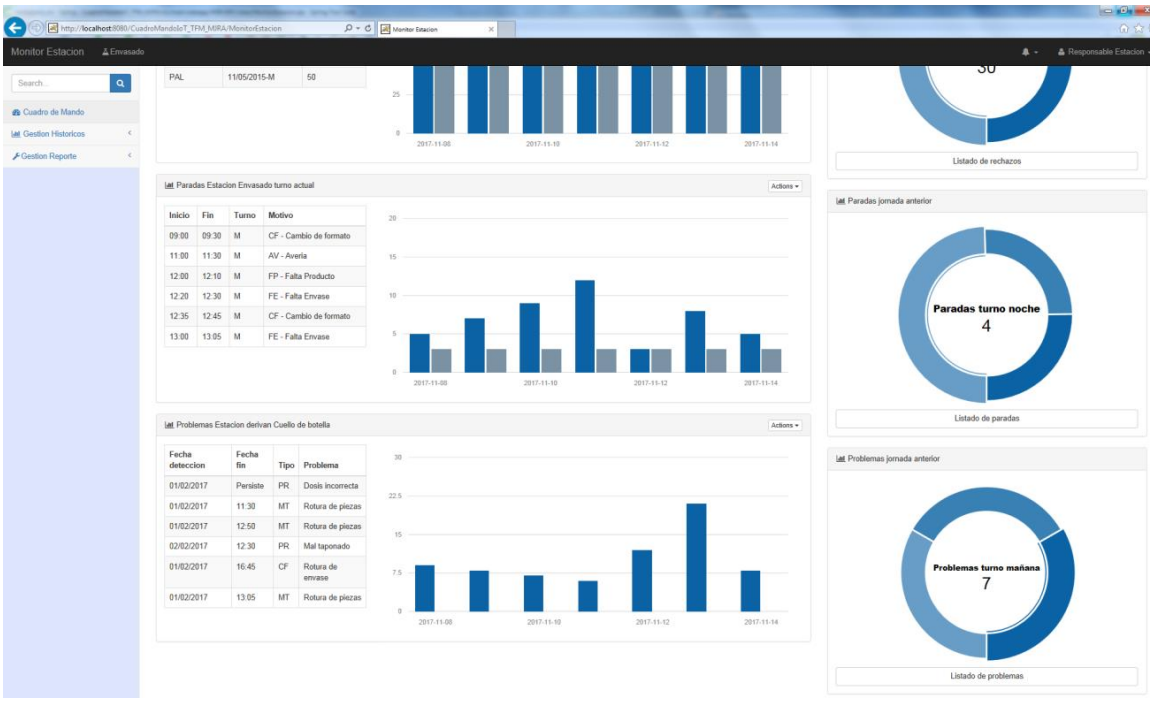


ILUSTRACIÓN 89: DETALLE GRÁFICO PARADAS Y PROBLEMAS DE ESTACIÓN MONITORIZADA

El detalle de los gráficos seguirá la misma línea de estilo que rechazos. Por un lado, mostrará el gráfico del volumen de paradas de la última semana y las paradas surgidas en la estación en turno:

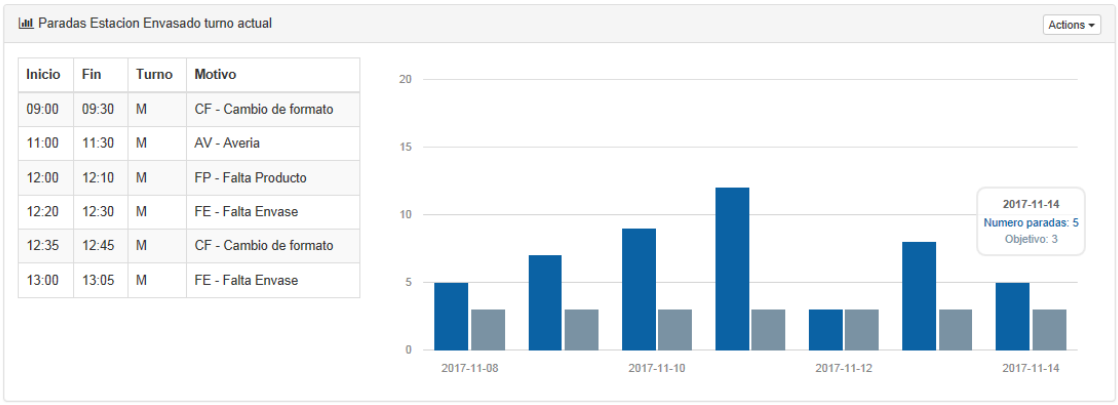


ILUSTRACIÓN 90: DETALLE GRÁFICDO ANÁLISIS DE PARADAS

Lo mismo en cuantos problemas surgidos:

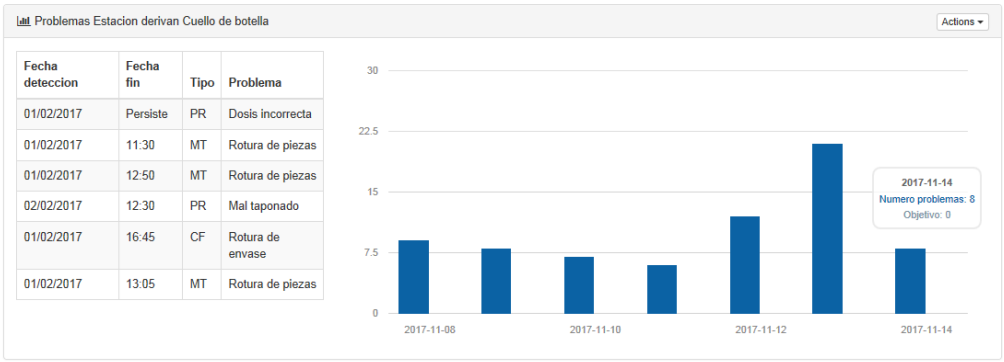


ILUSTRACIÓN 91: GRÁFICO ANÁLISIS PROBLEMAS

Los gráficos tipo rosca mostrará paradas y problemas de los últimos tres turnos respectivamente:

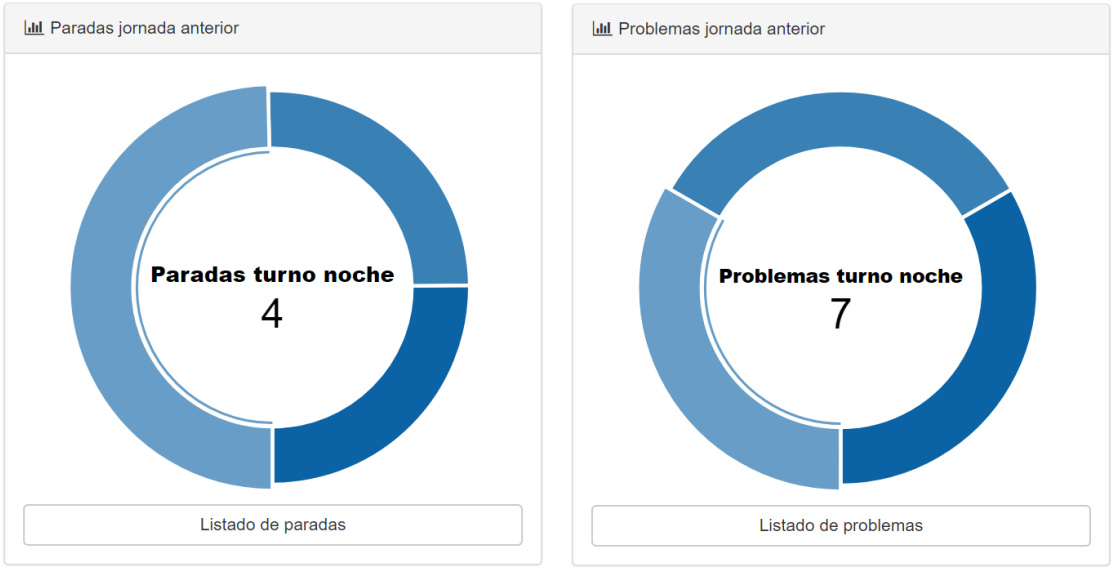


ILUSTRACIÓN 92: GRÁFICOS ROSCA PARADAS Y PROBLEMAS EN ESTACIÓN

Al ser un diseño de interfaz de usuario adaptado, se ha logrado que, ante un panel de mayores dimensiones, por ejemplo, un área que gestione un control de planta, se pueda visualizar el cuadro de mando al 100%:

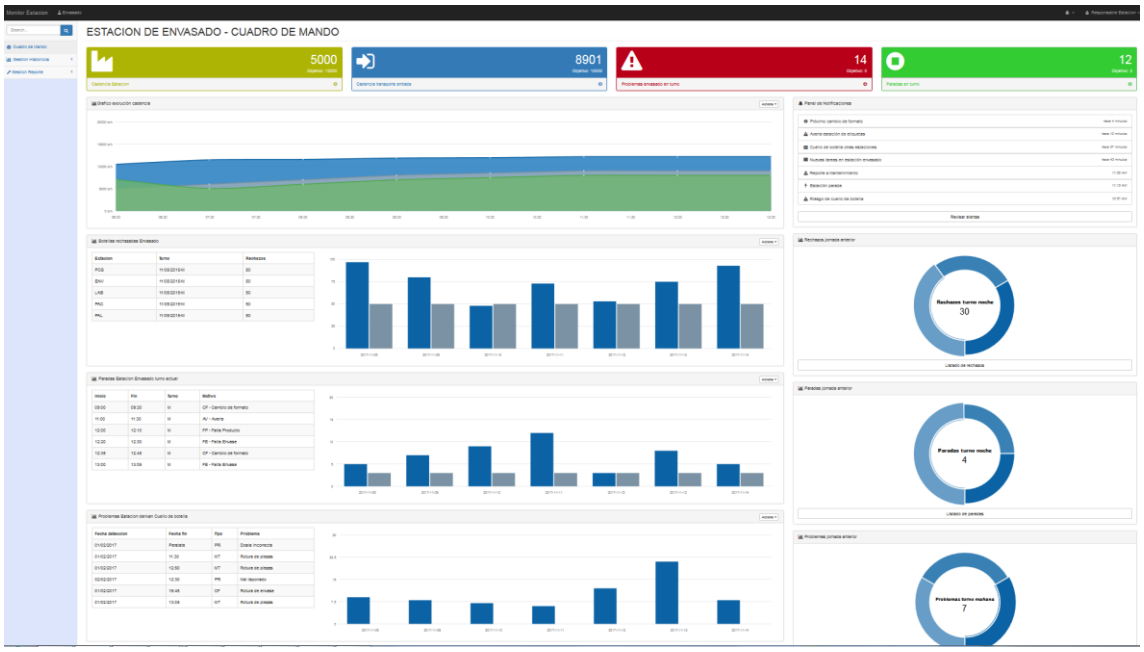


ILUSTRACIÓN 93: VISIÓN 100% PANTALLA DE MONITORIZACIÓN

4.1.2. GESTIÓN DE HISTÓRICOS

La siguiente interfaz diseñada permitirá visualizar gráficos a partir de una fecha seleccionada, el acceso a la funcionalidad se realizará desde el menú lateral:

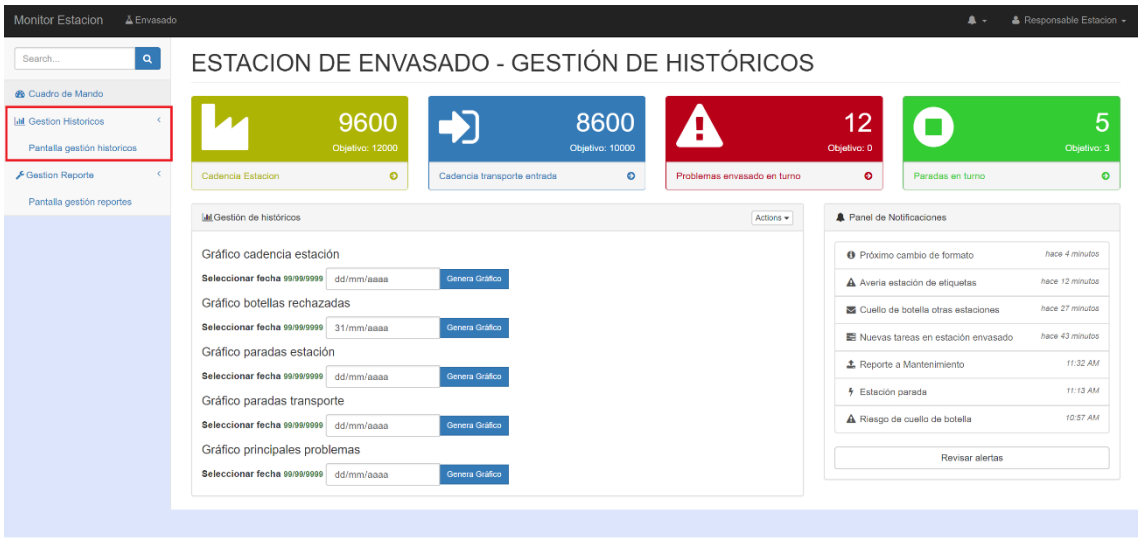


ILUSTRACIÓN 94: PANTALLA FUNCIONALIDAD GESTIÓN HISTÓRICOS

La interfaz incluye el encabezado con los datos clave de la estación para tenerlos monitorizados en cualquier momento y el panel de notificaciones.

- Detalle Gestión de históricos:

A partir de la idea vista en el diseño de las maquetas de prototipo, se configura una página que conseguirá acceder a los gráficos de las diferentes variables que permiten detectar cuellos de botella:

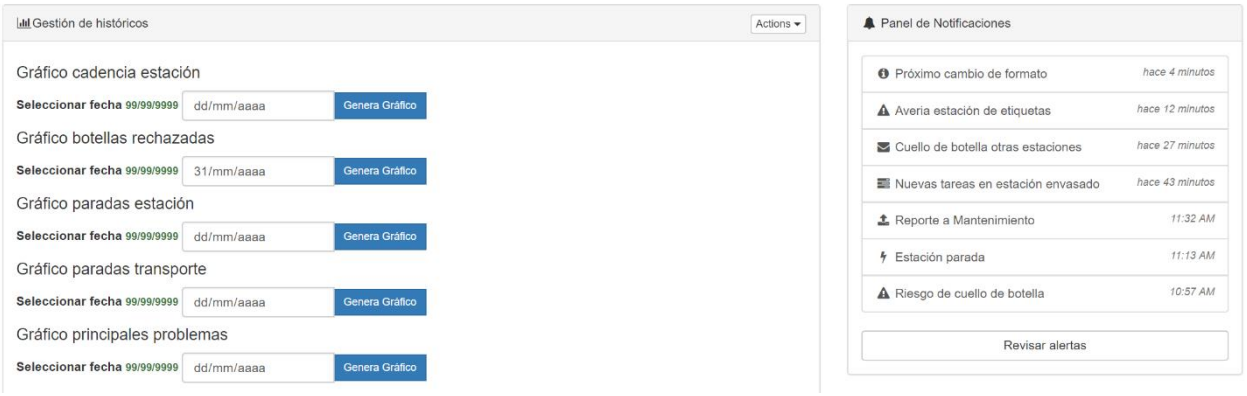


ILUSTRACIÓN 95: DETALLE VISTA GESTIÓN HISTÓRICOS

El modo de acceso será insertar la fecha de monitorización, en este sentido el sistema consultará su existencia:

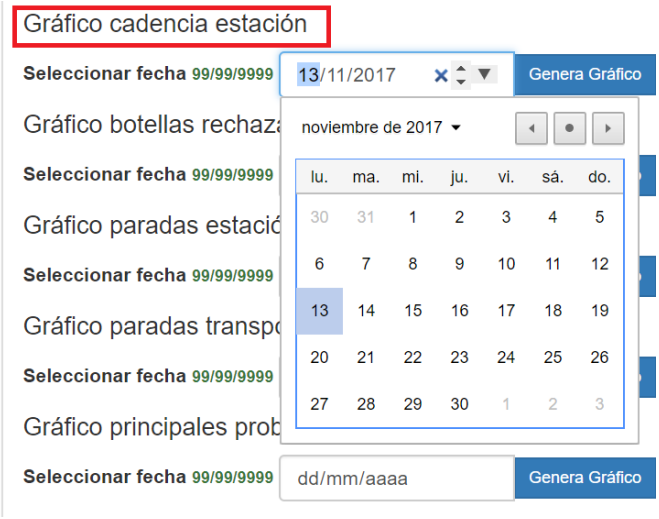


ILUSTRACIÓN 96: DETALLE SELECCIÓN FECHA Y GENERACIÓN GRÁFICO

Una vez verificado mostrará el resultado, en el caso del prototipo se acota a la visibilidad del gráfico de cadencia de estación en la fecha según la imagen, cuyo resultado será:

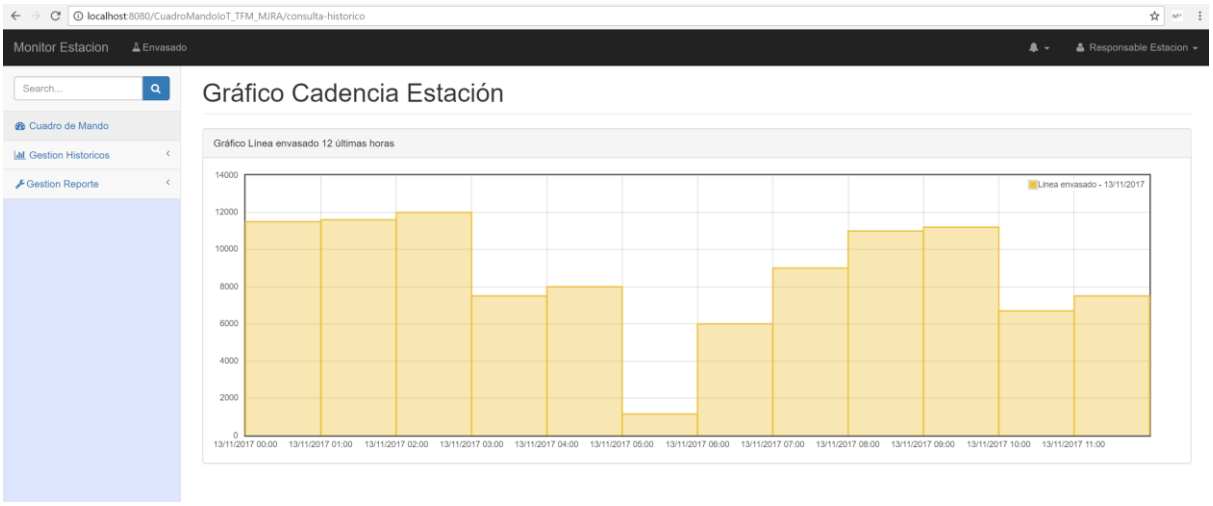


ILUSTRACIÓN 97: DETALLE GRÁFICO RESULTADO

El gráfico diseñado es de tipo barras y mostrará el detalle de la variable en las últimas 12 horas, situando el cursor sobre una de las barras la pantalla mostrará el siguiente detalle:

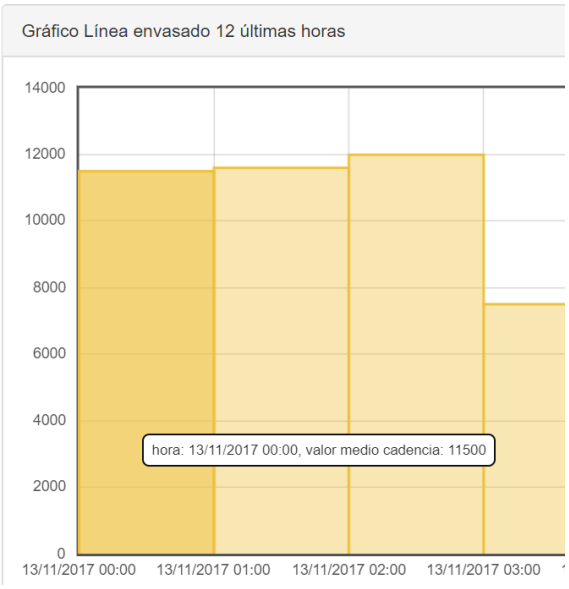


ILUSTRACIÓN 98: DETALLE AMPLIADO VISTA GRÁFICO HISTÓRICO

De esta forma se conseguirá poder examinar un histórico que el diseño visual permita monitorizarlo, obteniendo datos de las últimas 12 horas. Indicar que el

sistema es escalable y permitirá dinamizar los gráficos para ampliar o disminuir horas.

4.1.3. REPORTE ON LINE

La presente interfaz mostrará la siguiente vista, el acceso a la funcionalidad se realizará de la misma forma que el resto, desde el menú lateral:

ILUSTRACIÓN 99: DETALLE VISTA REPORTE

La interfaz seguirá la misma línea que la anterior, mostrará las variables importantes y el panel de notificaciones.

Su contenido específico para mostrar los reportes desde la aplicación online será como se indica en las siguientes líneas.

El sistema permitirá seleccionar informe según su tipología:

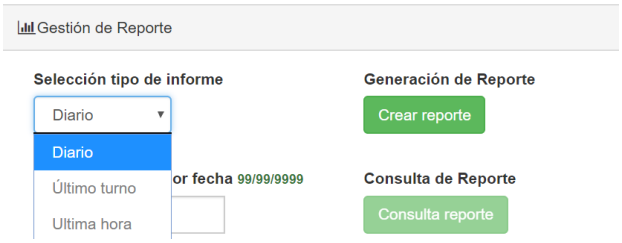


ILUSTRACIÓN 100: DETALLE AMPLIADO SELECCIÓN INFORME

En el caso del prototipo se emula un informe diario:

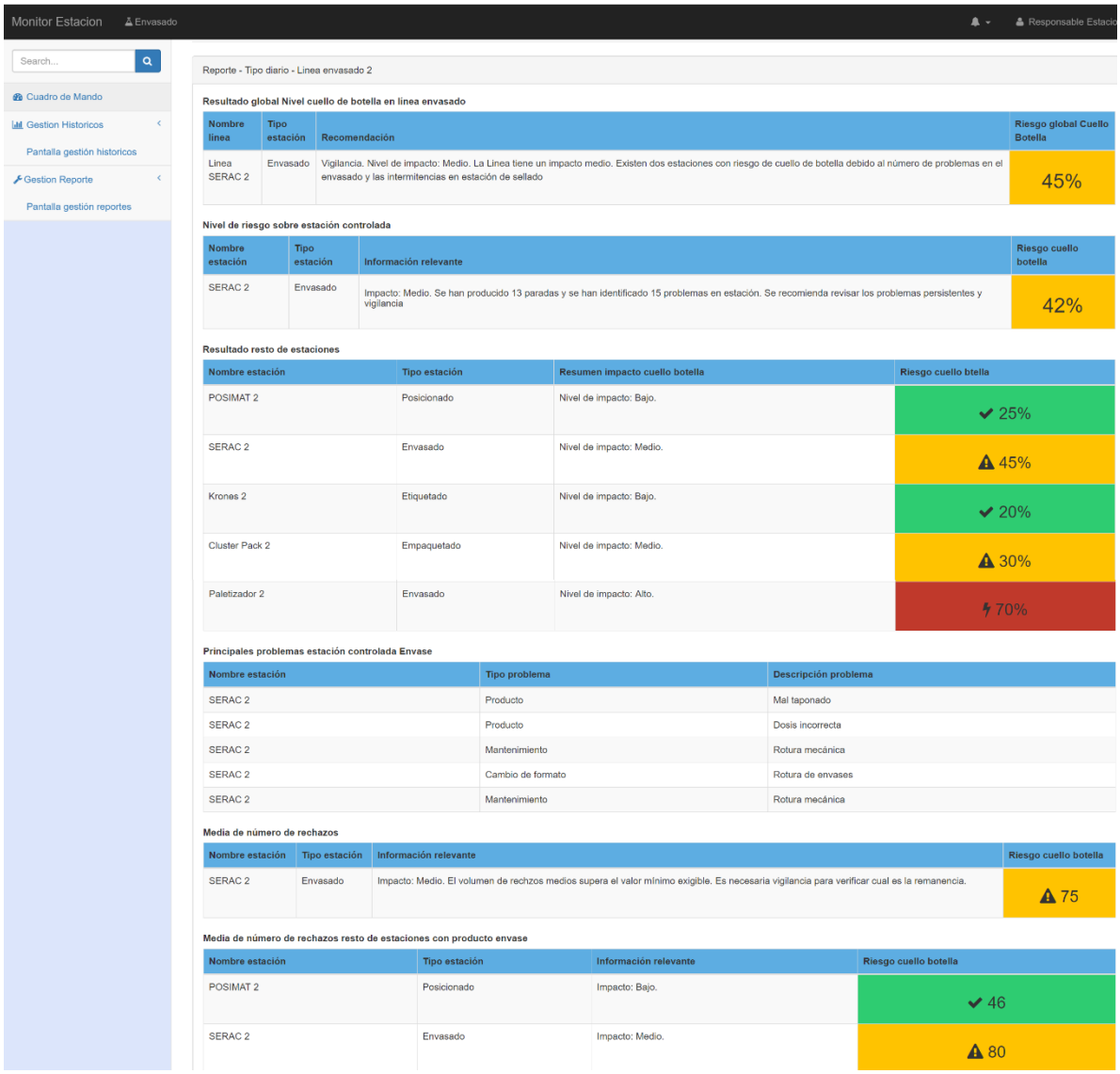
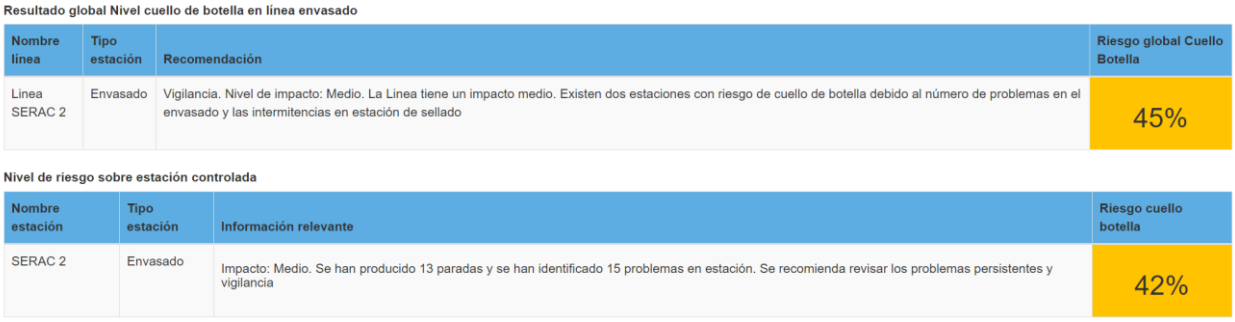


ILUSTRACIÓN 101: PROTOTIPO INFORME DIARIO GENERADO

El informe mostrará datos relevantes y un resultado con un grado de riesgo que permita dar información al usuario para tomar decisiones y actuar sobre la estación monitorizada. El prototipo emula datos a nivel de simulación con una iniciativa de recomendaciones y grado de riesgo de cuello de botella.

Con ello se consigue el objetivo inicial sobre la funcionalidad de reporte de obtener un reporte con información sobre problemática detectada de cuello de botella.

En una situación real se deberá de estudiar el caso de uso, los grados de riesgo dependerán del contexto de estudio de la línea de envasado y las necesidades de producción a partir de las bases investigadas en el presente trabajo.



Resultado global Nivel cuello de botella en línea envasado

Nombre línea	Tipo estación	Recomendación	Riesgo global Cuello Botella
Linea SERAC 2	Envasado	Vigilancia. Nivel de impacto: Medio. La Linea tiene un impacto medio. Existen dos estaciones con riesgo de cuello de botella debido al número de problemas en el envasado y las intermitencias en estación de sellado	45%

Nivel de riesgo sobre estación controlada

Nombre estación	Tipo estación	Información relevante	Riesgo cuello botella
SERAC 2	Envasado	Impacto: Medio. Se han producido 13 paradas y se han identificado 15 problemas en estación. Se recomienda revisar los problemas persistentes y vigilancia	42%

ILUSTRACIÓN 102: DETALLE BLOQUES ANÁLISIS RESULTADOS DEL REPORTE PROTOTIPO

5. CONCLUSIONES

El espíritu de este trabajo ha sido el mostrar los resultados de un estudio sobre los problemas que se pueden generar en cuanto a cuellos de botella en una línea de envasado y diseñar un sistema global basado en el paradigma IoT e industria 4.0 para poder ayudar a solucionar y pormenorizar las problemáticas a partir del análisis de los datos clave que provocarán las problemáticas.

Para ello se ha puesto en contexto sobre la situación actual en cuanto a los componentes de una línea de envasado y las principales problemáticas que se pueden generar en cuanto a cuellos de botella.

Se ha mostrado información de cómo se componen estas líneas, como se enlaza el producto fabricado entre estaciones, su gobierno, componentes clave en la automatización y otros aspectos para finalmente identificar y clasificar las problemáticas en las estaciones de trabajo acerca del cuello de botella.

se ha propuesto una idea para mejorar la situación a partir de un sistema global basado en el paradigma de la Industria 4.0

A partir de aquí, sobre la problemática que se quería tratar inicialmente obtenemos los siguientes puntos desarrollados en el trabajo. Diseñar un sistema con el objetivo de:

- Capturar la información relevante a partir de las variables susceptibles de generar un cuello de botella investigadas previamente.
- Monitorizar todas estas variables a partir de la división de una interfaz de usuario en bloques de información lógica.
- Crear funcionalidades sensibles para emitir información que permita tomar decisiones para subsanar la problemática de los cuellos de botella a partir de reportes e información de históricos.

- Crear un prototipo para mostrar ejemplares sobre las vistas en cuanto a la monitorización de las variables clave, reporte de información y gráfico con datos históricos.

Con los resultados de este sistema, los responsables del producto fabricado podrán anticiparse a cuellos de botella en el proceso de envasado y tomar decisiones para corregir los posibles desvíos medidos.

El sistema, además, permitirá poder enviar e intercambiar información relevante a otros sistemas a través de servicios, reportes de forma periódica.

En definitiva, los conocimientos previos vistos en las disciplinas a lo largo de los estudios y la dotación de conocimientos para la investigación han permitido diseñar un sistema con un punto de partida que analice y ayude a resolver problemáticas sobre cuellos de botella en líneas de envasado, de carácter global, además de adaptable y escalable a otras funciones.

6. LÍNEA DE TRABAJOS FUTUROS

Las conclusiones que ha dado el trabajo de investigación han permitido conocer y clasificar errores de cuello de botella de forma general en las estaciones de una línea de envasado, una variable clave a medir y la construcción de un sistema en base al análisis previo de las problemáticas.

Para la adaptación a un caso específico de línea de envasado se debería estudiar el caso de negocio y verificar que los aspectos diseñados inicialmente se adaptan correctamente a la situación real. El sistema diseñado es escalable y el módulo de integración utiliza estándares de comunicación para poder adaptarse a la comunicación con otros sistemas, por lo tanto, facilitará la adaptación.

Con todo esto, para implantar un caso de línea de futuro, la solución debería de tener en cuenta los siguientes aspectos:

- Estudio de implantación de los sensores de cadencia, entrada de estación y RFID, para este último se debería de evaluar si el envase cuenta con emisor o estudiar la posibilidad de uso de otros identificadores como código de barras para resolver su trazabilidad.
- Revisar el sistema de seguridad diseñado y adaptarlo a los sistemas del cliente además de dotar una autenticación adecuada a estándares como LDAP u otro protocolo de autenticación.
- La monitorización es un punto clave, por lo tanto, se tendría que verificar la necesidad de refresco de las variables medidas, en este diseño se especificó una simulación que dotaba inicialmente una frecuencia de 5 minutos.
- Revisar reporte Online y reporte Batch.
    - Implementar reglas basadas en la situación real de la línea de producción para crear los porcentajes reales de impacto y riesgo

de cuello de botella a la hora de comparar los datos y crear informes.

- Verificar si la frecuencia diaria, última hora y turno es eficaz sino adaptar al caso específico.

Otros puntos de trabajo de futuro sobre el presente sistema serían:

- Dotar/ampliar el sistema Web con un sinóptico con datos en tiempo real que verifique los puntos en los cuales se provoque un cuello de botella. Mostrando unos indicadores

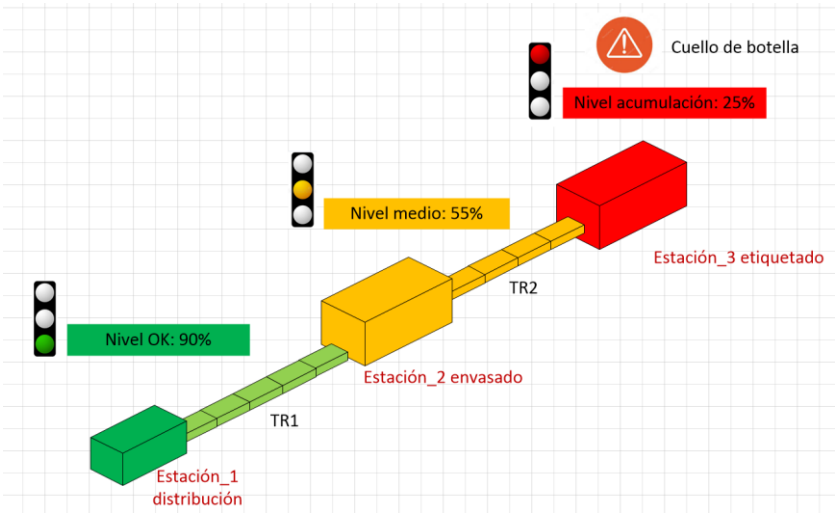


ILUSTRACIÓN 103: POSIBLE SINÓPTICO CON INDICADORES DINÁMICOS LÍNEA ENVASADO

- Utilizar inteligencia artificial para el autoaprendizaje en la línea de envasado de cara a detectar de forma más eficaz cuellos de botella.

7. BIBLIOGRAFÍA Y REFERENCIAS

[SIM01] Seven examples of a Bottleneck. Accedido desde: http://simplicable.com/new/bottleneck

[WIK01] El cuello de botella Accedido desde: https://es.wikipedia.org/wiki/Cuello_de_botella

[OPE01] Finding & Managing Bottlenecks in Process Plants. Accedido desde: https://opexsociety.org/body-of-knowledge/finding-managing-bottlenecks-in-process-plants/

[ATO01] Tipos en el manejo de cuellos de botella. Accedido desde:

http://www.pcs-info.com/article_tips_on_managing_bottlenecks.htm

[PLA01] PlantRun OEE, Machine Downtime & Manufacturing Information Systems. Accedido desde: http://plantrun.co.uk/managing-manufacturing-bottlenecks.html

[WIK02] Teoría de las limitaciones. Accedido desde: https://es.wikipedia.org/wiki/Teor%C3%ADa_de_las_limitaciones

[GES01] Teoría de Restricciones TOC - Theory of Constraints. Accedido desde: https://www.gestiopolis.com/teoria-de-restricciones-toc-theory-of-constraints/

[POS01] Posicionador de botellas. Accedido desde: http://posimat.com/index.php/es/productos/posicionadores/posicionadores/39-master/88-master

[WIK03] Packaging machines. Accedido desde:

https://en.wikipedia.org/wiki/Packaging_machine

[ENV01] Inline filling systems. Accedido desde:

http://www.fillers.com/envasadora/

[WIK04] Bottling line. Accedido desde:

https://en.wikipedia.org/wiki/Bottling_line#See_also

[SER01] Máquinas rotativas para envasado de yogurt. Accedido desde: http://www.dairy-bottle-packaging-machines.com/yogures-liquidos-y-leches-fermentados.html

[DAR01] Máquinas rotativas para el tratamiento individual de jeringas. Accedido desde: http://www.dara-pharma.com/es/pharma/58-llenado-cerrado/47-sfl-r-maquinas-rotativas-para-el-tratamiento-individual-de-jeringas.html

[SYS01] SYSMA, llenadoras rotativas y lineales. Accedido desde: http://www.sysma.com.mx/llenadoras.php

[ALT01] Altech, etiquetadoras lineales. Accedido desde: http://www.altech-la.com/esp/linear-labelling-machines.html

[KRO01] Etiquetado rotativo a través de Krones canmatic. Accedido desde: https://www.krones.com/es/productos/maquinas/etiquetado-envolvente-con-etiquetas-precortadas.php

[WST01] Cluster-pack® Multipack systems. Accedido desde: https://www.westrock.com/en/products/folding-cartons/cluster-pack-multipack-system

[TEC01] Grupo teckmapack. Accedido desde: http://www.tecma-pack.fr/index-es.php

[KRO02] Krones variopac PRO. Accedido desde: https://www.krones.com/es/productos/maquinas/embaladora-variable.php

https://www.krones.com/media/downloads/VariopacPro_es.pdf

[ULM01] ULMA, sistemas de paletizados. Accedido desde: http://www.ulmapackaging.com/maquinas-de-envasado/soluciones-integrales/sistemas-de-paletizado

[WIK05] Palé, Accedido desde: https://es.wikipedia.org/wiki/Pal%C3%A9

[PLC01] Autómata programable. Accedido desde: https://en.wikipedia.org/wiki/Programmable_logic_controller

[HRD01] Hardware libre. Accedido desde: https://es.wikipedia.org/wiki/Hardware_libre

[MOT01] Motores de inducción polifásicos "Tesla polyphase induction motors". Accedido desde: https://www.ibiblio.org/kuphaldt/electricCircuits/AC/AC_13.html#xtocid353418

[DIN01] Dinamos tacométricas Accedido desde: http://rccindustrial.com/dinamos-tacometricas/

[WIK06] Finales de carrera. Accedido desde: https://es.wikipedia.org/wiki/Sensor_final_de_carrera

[ABB01] Finales de carrera ABB. Accedido desde: http://new.abb.com/low-voltage/es/productos/interruptores-de-final-de-carrera

[WIK07] Rotary encoder. Accedido desde: https://en.wikipedia.org/wiki/Rotary_encoder

[WIK08] Detector inductivo. Accedido desde: https://es.wikipedia.org/wiki/Sensor_inductivo

[KEY01] ¿Qué es un detector de proximidad inductivo? Accedido desde: http://www.keyence.com.mx/ss/products/sensor/sensorbasics/proximity/info/

[FOU01] Las corrientes Foucault. Accedido desde: http://www.sc.ehu.es/sbweb/fisica_/elecmagnet/faraday/foucault/foucault.html

[WIK09] Sensor de efecto hall. Accedido desde: https://es.wikipedia.org/wiki/Sensor_de_efecto_Hall

[WIK10] Sensor fotoeléctrico. Accedido desde: https://es.wikipedia.org/wiki/Sensor_fotoel%C3%A9ctrico

[KEY02] ¿Qué es un sensor ultrasónico? Accedido desde: http://www.keyence.com.mx/ss/products/sensor/sensorbasics/ultrasonic/info/

[ROB01] Sensor de ultrasonidos. Accedido desde: http://wiki.robotica.webs.upv.es/wiki-de-robotica/sensores/sensores-proximidad/sensor-de-ultrasonidos/

[PEF01] pepperl-fuchs. Accedido desde: http://www.pepperl-fuchs.es

[ALB01]. Sensores fotoeléctricos de Allen Bradley Accedido desde: http://ab.rockwellautomation.com/es/Sensors-Switches/Photoelectric-Sensors

[IIA01] Fundamentals of Industrial Instrumentation and process control. William C. Dun, Mc. Graw Hill.

[WIK10] El termopar. Accedido desde: https://es.wikipedia.org/wiki/Termopar

[IEC01] Normativa IEC. Accedido desde: https://webstore.iec.ch/publication/4550

[PC001] SIDE PC industrial. Accedido desde: http://www.side-automatizacion.com/es/pc-industrial

[PC002] Omron PC Industrial. Accedido desde: https://industrial.omron.es/es/products/industrial-pc

[IEC02] Introducción al estándar IEC-61131. Accedido desde: http://isa.uniovi.es/docencia/IngdeAutom/transparencias/Pres%20IEC%2061131.pdf

[UVA01] PC Industriales. Accedido desde: http://www.uv.es/rosado/courses/sid/Capitulo5_PCind.pdf

[AER01] Transporte aéreo de botellas. Accedido desde: http://posimat.com/es/productos/transportadores/posijet

[COM01] Máquina de envasado comatec. Accedido desde: http://www.comatecsa.com/productos/familias/7/Envasado+de+leche

[KHS01] Máquina de etiquetado KHS. Accedido desde: http://www.directindustry.es/prod/khs-gmbh/product-21322-1828928.html

[CAM01] Video proceso de empaquetado y paletizado por robot. Accedido desde: http://www.directindustry.es/prod/cama-group/product-56046-528792.html

[UPM01] - Tiempos de fabricación. ETSII – UPM. Accedido desde: http://wikifab.dimf.etsii.upm.es/wikifab/images/9/96/4._Tiempos_de_fabricaci%C3%B3n.pdf

[AOP01] Aspect-oriented programming. Accedido desde: https://en.wikipedia.org/wiki/Aspect-oriented_programming

[AOP02] Aspect Oriented Programming with Spring 11.1.1 AOP concepts. Accedido desde: https://docs.spring.io/spring/docs/current/spring-framework-reference/html/aop.html

[ARD01] Overview de Arduino Yun. Accedido desde: https://store.arduino.cc/arduino-yun

[MIC01] Sensor ultrasónico modelo mic+. Accedido desde: https://www.microsonic.de/es/sensores-de-distancia/cilindrico/micplus/sensores-est%C3%A1ndar/sensores-est%C3%A1ndar/micplus340iutc.htm

[SPR01] Módulo Spring modelo vista controlador. Accedido desde: https://docs.spring.io/spring/docs/current/spring-framework-reference/web.html

[SPR02] Framework Spring. Accedido desde: https://spring.io/

[SPR03] Microservicios a través de Spring boot:. Accedido desde: https://projects.spring.io/spring-boot/

[BOO01] Bootstrap: open source toolkit for developing with HTML, CSS, and JS. Accedido desde: https://getbootstrap.com/

[TOM01] Contenedor de servlets Tomcat. Accedido desde: http://tomcat.apache.org/

[SIE01] Mindsphere Application Center. Accedido desde: https://www.siemens.com/global/en/home/products/software/mindsphere.html

[SCH01] Servicios en la nube de Schneider. Accedido desde: https://www.schneider-electric.es/es/work/services/cloud/#

[ROC01] Conjunto de aplicaciones IoT Rockwell en Azure. Accedido desde: https://www.microsoft.com/es-es/cloud-platform/customer-stories-rockwell-automation

[MIC01] Clientes de Microsoft Azure y Azure IoT. Accedido desde: https://customers.microsoft.com/en-us/search?sq=%22Azure%20IoT%20Hub%22&ff=&p=0&so=story_publish_date%20desc

[KAA01] Plataforma Open Source Kaa. Accedido desde: https://www.kaaproject.org/

[SIT01] Plataforma IoT Sitwhere. Accedido desde: http://www.sitewhere.org/

8. DEFINICIÓN DE SIGLAS, ABREVIATURAS Y ACRÓNIMOS

Término	Definición
http	*Es el protocolo de comunicación a nivel de aplicación que permite el intercambio de información entre sistemas por la web. http define la sintaxis y semántica que utilizan los elementos software de la arquitectura como servidores, clientes, proxys.*
BO	*business object. Pertenece al alcance de objetos de negocio que interactúan entre la capa de presentación y la capa de datos dentro de un modelo de diseño de tres capas para implementar aplicaciones bajo el paradigma web basadas en lenguajes orientados a objetos.*
DAO	*Data Acces Object. Dominio objetos donde se implementa la lógica del modelo de persistencia de datos. Representa la capa más baja dentro del modelo de diseño en tres capas. Estos objetos conectan con el sistema de gestor de base de datos para realizar operaciones contra esta.*
J2EE	*Es una plataforma de programación del lenguaje Java para implementar desarrollos empresariales. permite la programación en el paradigma de N capas, cliente servidor, servicios. Contiene librerías con especificaciones para Servicios web, mensajería con colas, jdbc, servlets, JSP, etc.*
IOT	*Internet of Things. Concepto referido a la interconexión de objetos o elementos que permiten*

	tener conectividad a través de internet. Por ejemplo, el paradigma se basa en casos como el específico de la industria 4.0. Por ejemplo, para tener acceso a la información de sensores desde otro punto deslocalizado. En este caso específico se precisará establecer seguridad digital para controlar el acceso a la información.
MQTT	*MQTT es un protocolo de conectividad de máquina a máquina (M2M) / "Internet de las cosas". Fue diseñado como un transporte de mensajería a través de un bróker intermedio para publicar y suscribirse a información ligera.*
REST	*Traducido como transferencia de estado representacional, es un estilo de diseño para sistemas distribuidos como servicios bajo el paradigma Web. Como base consta de una serie de instrucciones simples para operaciones CRUD. Basado en el paradigma cliente-servidor, permite publicar servicios web ligeros sin estado.*
JSON	*Notación de objetos Javascript, es un formato de texto ligero para el intercambio de datos a través de caracteres delimitados. Se presenta como una alternativa al intercambio de datos con XML. Presenta un analizador sintáctico más simple que los parseadores de XML.*
P2P	*Paradigma de comunicación por pares. La comunicación es por igual entre dos sistemas. Permite intercambiar información actuando los dos simultáneamente con los roles de cliente y servidor.*
UDP	*es un protocolo del nivel de transporte basado en el intercambio de datagramas (Encapsulado de capa 4 o de Transporte del Modelo OSI). Permite el envío*

	de datagramas a través de la red sin que se haya establecido previamente una conexión, ya que el propio datagrama incorpora suficiente información de direccionamiento en su cabecera.
OPC	*Es un estándar de comunicación en el campo del control y supervisión de procesos industriales, basado en una tecnología Microsoft, que ofrece una interfaz común para comunicación que permite que componentes de software individuales interactúen y compartan datos.*
NOSQL	*Es una clase de Sistema de gestión de base de datos que difiere del modelo relacional cuyo aspecto principal es que no utiliza lenguajes de consultas y los datos no precisan estructuras en tablas. Principalmente se utiliza su almacenamiento en clave-valor, pero pueden contener estructuras bajo paradigmas como el almacenamiento documental, en forma de grafo, multivalor, tabular, en estructuras de arrays, etc., …*
SQL	*Es un lenguaje específico del dominio que da acceso a un sistema de gestión de bases de datos relacionales que permite especificar diversos tipos de operaciones en ellos.*
PLC	*Es un sistema electrónico basado en microprocesador utilizado principalmente en las áreas de automática o automatización industrial para el control de maquinaria, líneas y procesos industriales, su lenguaje está preparado para llevar a cabo una programación basada en estados secuenciales y procesar datos tanto analógico como digitales. Está normalizado bajo el estándar IEC 61131-3.*

MVC	*El modelo vista controlador es un patrón arquitectónico que separa la lógica de negocio de la presentación. El controlador permitirá recoger las peticiones de la vista en el lado cliente y resolver la lógica demandada en el lado servidor a través del modelo. Este a su vez, responderá al controlador con los datos precisos que requería la vista o retornará un error si no se han podido resolver.*

9. ANEXOS

ANEXO 1. TIPOS DE MÁQUINAS DE ENVASADO LINEALES

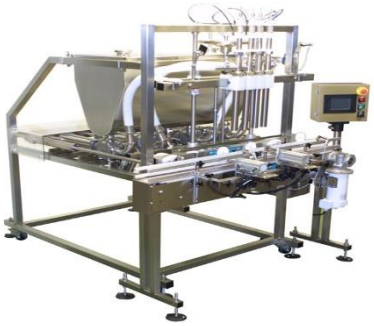
Tipos de máquinas lineales según el fabricante [ENV01]

LLENADORA POR SOBRECARGA: Su diseño se orienta para líquidos con poca viscosidad y espuma.

LLENADORA SERVO BOMBA: Estilo bomba de alta tecnología y versatilidad para casi todos productos.

LLENADORA POR BOMBA PERISTALTICA: Se construye para líquidos poco viscosos, poco volumen de llenado y preciso. Por ejemplo, los aromas necesarios para los yogures de sabores.

LLENADORA POR GRAVEDAD: Este tipo de máquinas se utilizan para líquidos sin espuma y poco viscosos.



LLENADORA POR CILINDROS: Ideal para líquidos viscosos pero poca versatilidad. El líquido es insertado por accionamiento neumático, no tienen la misma precisión que las accionadas por bomba peristáltica.

LLENADORA PESO NETO: Llenado por peso, se construye para alto volumen de llenado y alta precisión.

LLENADORA LOCALIZACIÓN DE LÍQUIDOS INFLAMABLES: Especial para líquidos inflamable o explosivos.

Sistemas de Etiquetado Lineal

Los sistemas de etiquetado ALline se pueden configurar para el etiquetado de productos cilíndricos, elípticos y rectangulares o de formas irregulares, también para la aplicación de sellos de seguridad.

ALline C - Máquina Etiquetadora para Productos Cilíndricos

El sistema de etiquetado ALline C es ideal para la aplicación de etiquetas envolventes de productos cilíndricos, por lo general alimentarios, químicos, cosméticos y productos farmacéuticos, en plástico, vidrio o metal botellas, frascos.

ALline E - Sistema de Etiquetado para la Aplicación Frontal / Posterior

El sistema de etiquetado ALline E ha sido proyectado para aplicar uno, dos o más etiquetas frontales / posterior sobre productos elípticos o rectangulares,

generalmente productos alimentarios, cosméticos, productos farmacéuticos y productos químicos en plástico, botellas de vidrio o de metal y tarros.

ALline Special - Sistema de Etiquetado Múltiple, Aplicaciones en Angulo

El sistema de etiquetado ALline Special puede ser configurado para responder a requisitos específicos de etiquetado, como el etiquetado múltiple, en ángulo en productos de formas irregulares o inestables.

ALbelt - Sistema de Etiquetado Económico

ALbelt es una máquina etiquetadora lineal simplificada abierta y compacta, completamente automática y económica. Puede ser configurada para el etiquetado superior y lateral de varios productos, principalmente aquellos con superficies planas.

ALbelt C - Sistema de Etiquetado Envolvente Económico

ALbelt C es una máquina etiquetadora lineal compacta para la aplicación de etiquetas autoadhesivas envolventes sobre productos cilíndricos tales como botellas, latas, frascos y potes.

ALbelt dos Secciones - Sistemas de Etiquetado por Arriba y/o Abajo o C-Wrap

El sistema consiste en un transportador con dos secciones y puede etiquetar desde arriba y/o abajo o de canto en forma de U. Pueden ser utilizados uno o más aplicadores de la serie ALstep y ALritma, que a menudo son equipados con impresoras.

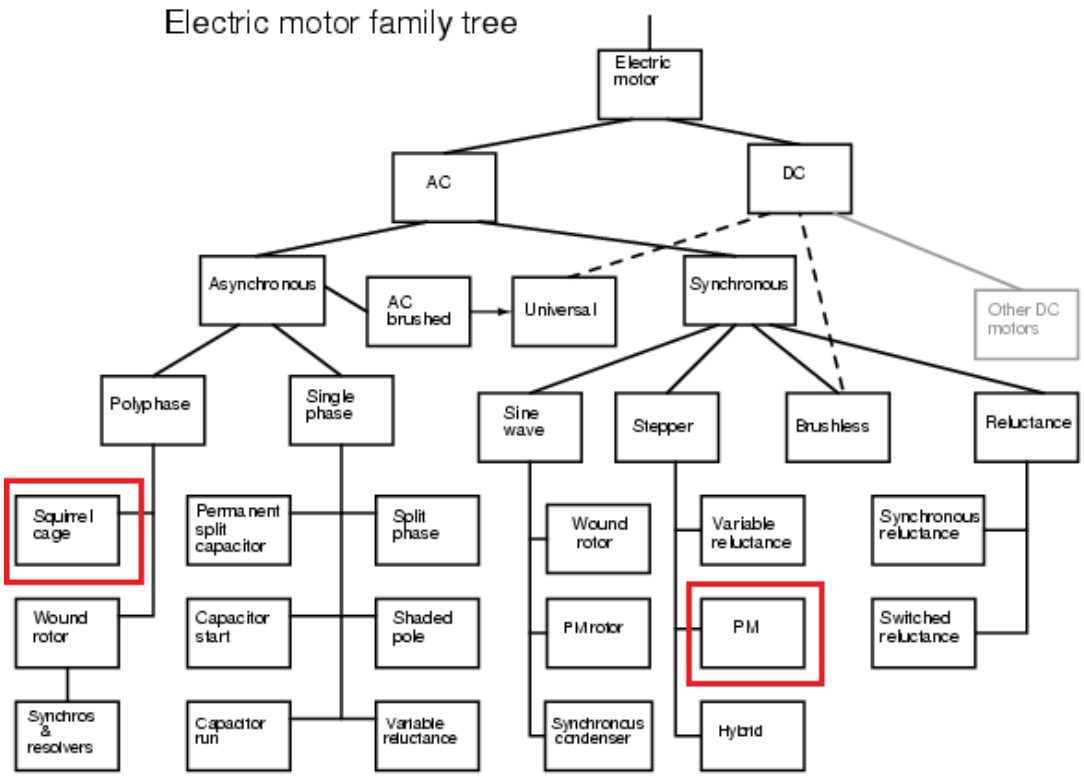


ILUSTRACIÓN 104: ÁRBOL FAMILIA DE MOTORES

En el siguiente árbol se muestra la familia de motores eléctricos tanto para corriente alterna como para corriente continua.

Dado que el alcance de este trabajo no está orientado al estudio detallado en motores, se fijarán dos tipos de motores más utilizados en la industria, según están marcados en la imagen. El motor asíncrono trifásico de rotor de jaula de ardillas y el motor paso a paso de imanes permanentes.

¿Por qué estos dos motores?

El primero, el llamado de rotor de jaula de ardillas o conocido como motor de inducción, es uno de los motores más utilizados para la tracción en cintas de

transporte de envasado en diferentes industrias como la alimentaria, farmacéutica o química.

El segundo, motor de imanes permanentes es porqué es el mayor utilizado para movimientos de precisión como el posicionamiento de cadenas o cintas en estaciones de trabajo, robótica, máquinas herramientas, ascensores etc.

Motores de inducción con rotor de jaula de ardillas

En primer lugar, este motor es llamado de jaula de ardillas por la similitud del rotor a una jaula de ardillas:

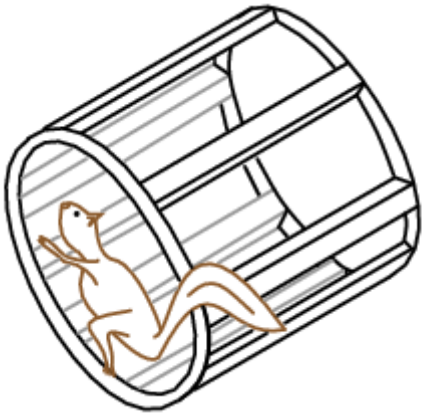


ILUSTRACIÓN 105: JAULA DE ARDILLAS (SIMILITUD A UN ROTOR)

Según [MOT01], el 90% de motores industriales son de inducción, estos son favorecidos por su robustez y simplicidad.

Como construcción un motor de dos fases tiene dos pares de bobinas un par para cada una de las dos fases de corriente alterna. Las bobinas individuales están conectadas en serie y corresponden a los polos opuestos de un electroimán.

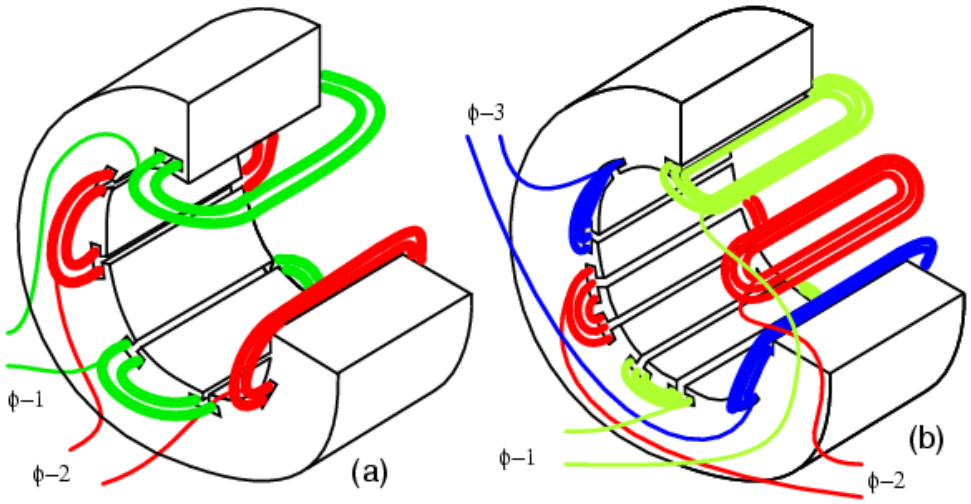


ILUSTRACIÓN 106: DEVANADOS MOTOR INDUCCIÓN

El rotor como se ha indicado tiene una similitud a una jaula de ardillas, es un eje donde va laminado de acero y una jaula de ardillas incrustada:

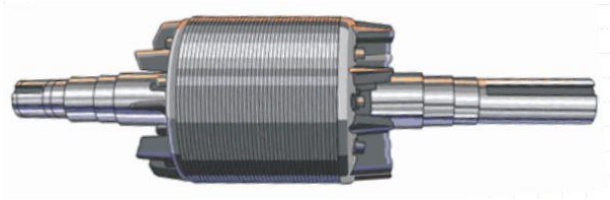


ILUSTRACIÓN 107: ROTOR JAULA DE ARDILLAS

Si se quiere estudiar con más detalles se puede investigar en [MOT01] donde se definen las diferentes derivaciones de los motores de inducción.

El uso de este tipo de motor está muy extendido en la industria y tal como se ha dicho se puede utilizar para la tracción de transporte de producto como para cualquier utilidad secundaria como bomba para transporte de fluidos, ventilación, etc. En algunos casos estos motores podrán trabajar en vacío y en otros con un reductor mecánico.

La variable que más se suele controlar en este tipo de motores es la velocidad a través de convertidores de frecuencia y menos común por la conmutación de polos.

ILUSTRACIÓN 108: CINTAS DE TRANSPORTE MOTORES INDUCCIÓN

Motores de imanes permanentes

Según los principales fabricantes como ABB o SIEMENS, Este tipo de motor tiene un rotor de imanes permanentes cilíndrico. El estator normalmente tiene dos bobinados. Los devanados pueden ser accionados centralmente para permitir un circuito de accionamiento unipolar en el que la polaridad del campo magnético se cambia conmutando una tensión de un extremo al otro del devanado.

Para alimentar a los devanados, se requiere un accionamiento bipolar de polaridad alterna auxiliar. Un paso magnético permanente puro tiene generalmente un ángulo grande. Por otro lado, La rotación del eje de un motor no excitado presenta par de enclavamiento.

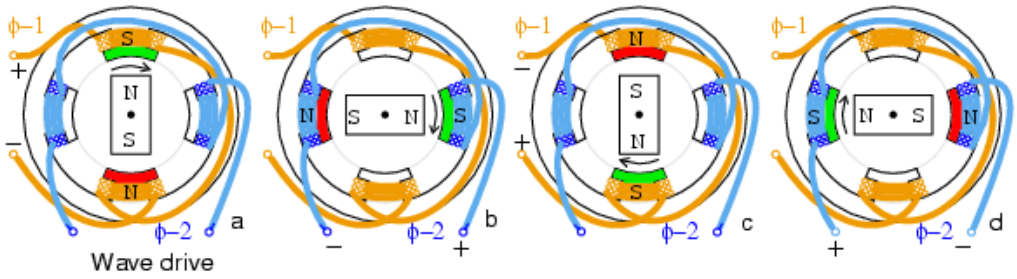


ILUSTRACIÓN 109: INDUCTOR E INDUCIDO DEL MOTOR IMANES PERMANENTES

Los motores paso a paso de imanes permanentes requieren corrientes alternas en fase aplicadas a los dos (o más) devanados. Se pueden aplicar casi siempre ondas cuadradas generadas de un componente de estado sólido. El accionamiento bipolar de ondas cuadrada puede alternar entre las polaridades (+) y (-), por ejemplo, +-5 voltios. El accionamiento unipolar suministra un flujo magnético alternativo (+) y (-) a las bobinas desarrolladas a partir de un par de cuadrados positivos Ondas aplicadas a los extremos opuestos de una bobina tachada central. La sincronización de la onda bipolar o unipolar es impulsión de la onda, paso completo, o medio paso.

Tal y como se ha mencionado arriba, este tipo de motor se utiliza para tener movimientos precisos, es por esto por lo que se utiliza sobre todo cuando es necesario un posicionamiento absoluto, como por ejemplo un robot.

Las variables que más se controlan en este tipo de motores son:

- Velocidad.
- Par motor.
- Posición.
- Temperatura.
- Frenado

A diferencia de los motores de inducción (aunque también pueden darse casos). Por norma general, este tipo de motor presenta un control en lazo cerrado, donde el bucle genera una realimentación sabiendo siempre la posición del

motor. Es por esto por lo que en la industria este tipo de motores viene compactado con una serie de controles de las variables anteriores tal y como se muestra en la siguiente figura:

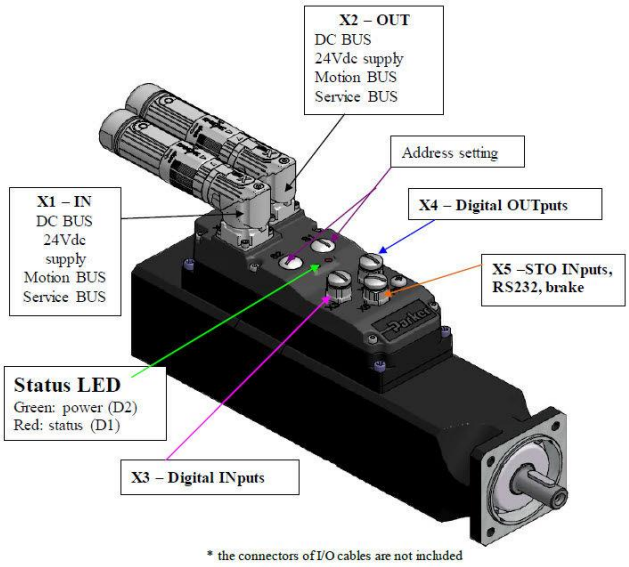


ILUSTRACIÓN 110: SERVORMOTOR DE IMANES PERMANENTES

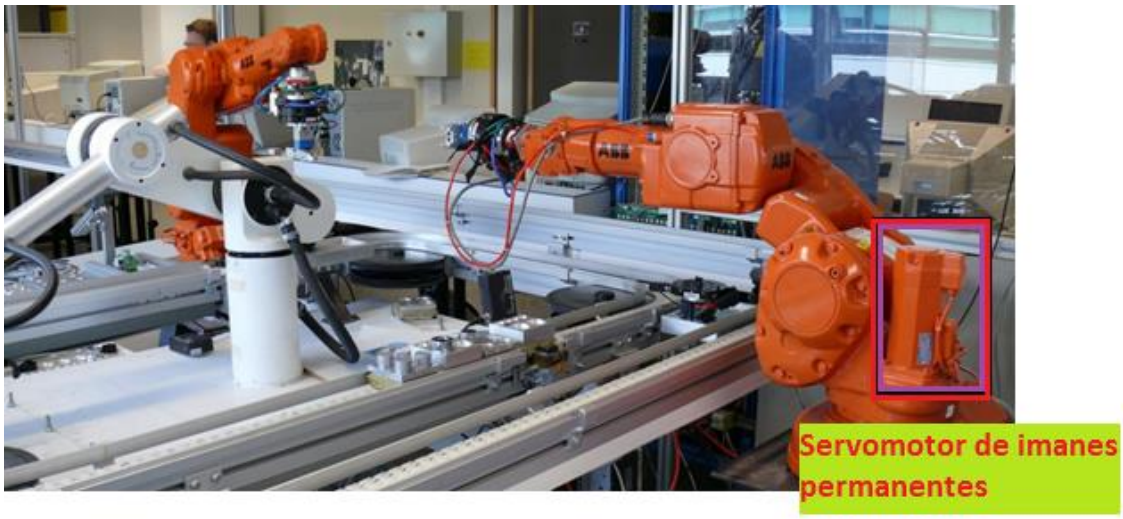
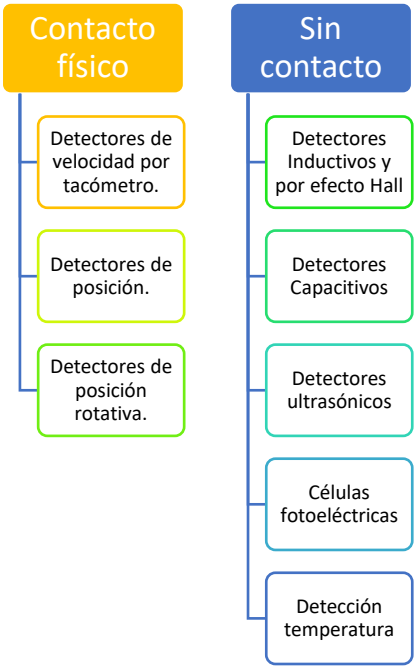


ILUSTRACIÓN 111: SERVO EN ROBOT ABB. © WWW.GOTRONIC.CO.UK

Dado el amplio espectro para analizar los diferentes sensores, existen muchos tipos de clasificación (por magnitud, por característica tecnológica, por naturaleza física, por contacto o lo contrario). Para este trabajo particular se ha llevado a cabo un análisis de los detectores que pueden influir en los motores para capturar información.

Para ello, en primer lugar, se realiza una clasificación por los detectores que pueden tener o no contacto sobre las máquinas.



Tal y como se puede ver, los detectores de contacto físico están formados por cuatro grandes grupos.

En las siguientes líneas se realiza una breve descripción sobre el estado de los sensores que tienen un acoplamiento físico con los motores:

DINAMO TACOMÉTRICA

En primer lugar, están los detectores de velocidad por dinamo tacométrica, según [DIN01]:

ILUSTRACIÓN 112: DINAMO TACOMÉTRICA

Las dinamos tacométricas son sensores eléctricos rotativos especiales que, en los últimos años, se han convertido en excelentes colaboradores en los procesos de regulación de velocidades de giro. La primitiva función de las dinamos tacométricas, denominadas también tacodinamos o tacómetros, fue simplemente de control, como indicadores del número de r. p. m. de ejes de giro. Actualmente sólo determinados tipos de tacodinamos, los más reducidos, simples y menos precisos, se utilizan como indicadores. La importancia de las dinamos tacométricas modernas reside en su participación directa en el proceso de regulación.

Sin entrar en detalles constructivos, es una máquina de corriente continua que trabaja como generador de tensión que transmite una señal en pulsaciones según la velocidad angular medida del motor acoplado lo cual permite obtener un rango de tensión en concordancia a la velocidad medida, esta señal se podrá acondicionar para obtener los datos medibles para el tratamiento.

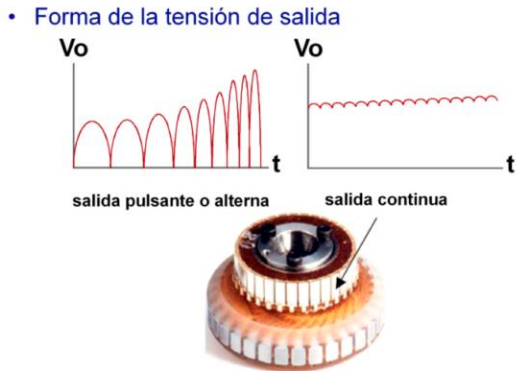


ILUSTRACIÓN 113: SEÑAL DE SALIDA TACODINAMO

Cabe indicar, que, al estar unido al eje de un motor, estos detectores sufren por calentamiento, además de padecer las consecuencias de las vibraciones que puede provocar el motor acoplado. Por esto motivo, provocará que sea más volátil que otro tipo de sensores que no tienen contacto directo.

Más abajo, se verá el detector rotativo, el cual puede proporcionar posición de un eje de motor, pero también puede ser utilizado para medir las revoluciones por minutos de un motor, teniendo en cuenta la velocidad angular del motor.

DETECTORES DE POSICIÓN POR CONTACTO

Este tipo de sensores son por contacto conmutado, pueden medir una posición del motor a través de su paso.

Su mecanismo es simple, trasmite una señal digital todo/nada. Por otro lado, según su construcción, los elementos pueden estar por defecto en posición (Normalmente cerrados) todo o al contrario nada (normalmente abiertos), y cambiar de estado al contactar con el recorrido del motor hasta una posición. Por norma general el eje del motor no contacta directamente con este sensor dado que un eje concéntrico no permitirá poder cambiar el estado del sensor.

Para poder cambiar el estado se realizará a través de una leva acoplada al eje de un motor o el movimiento transmitido a un eje secundario como por ejemplo una cinta de transporte o una persiana de un acceso.

Estos detectores son también conocidos como “finales de carrera”.

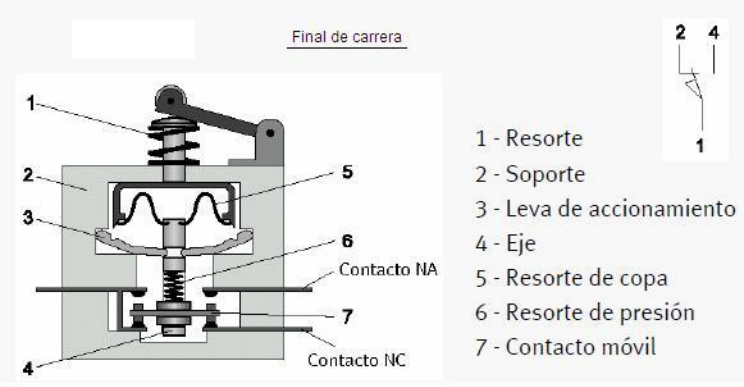


ILUSTRACIÓN 114: CONSTRUCCIÓN FINAL DE CARRERA

Según [WIK06] los define como:

Dentro de los componentes electrónicos, se encuentra el final de carrera o sensor de contacto (también conocido como "interruptor de límite"), son dispositivos eléctricos, neumáticos o mecánicos situados al final del recorrido o de un elemento móvil, como por ejemplo una cinta transportadora, con el objetivo de enviar señales que puedan modificar el estado de un circuito.

ILUSTRACIÓN 115: GAMA DE FINALES DE CARRERA DE © ABB [ABB01]

DETECTORES DE POSICIÓN ROTATIVA POR CONTACTO

Según [WIK07] el encoder o codificador rotario que definido como:

Un codificador rotatorio, también llamado codificador del eje o generador de pulsos, suele ser un dispositivo electromecánico usado para convertir la posición angular de un eje a un código digital, lo que lo convierte en una clase de transductor. Estos dispositivos se utilizan en robótica, en lentes fotográficas de última generación, en dispositivos de entrada de ordenador (tales como el ratón y el trackball), y en plataformas de radar rotatorias. Hay dos tipos principales: absoluto e incremental (relativo).

A esta definición, cabría añadir que este tipo de detector es utilizado en cualquier situación donde se requiera tener un control sobre la posición de un eje accionado por un motor.

ILUSTRACIÓN 116: CONTROL POSICIÓN ANGULAR DE RATÓN

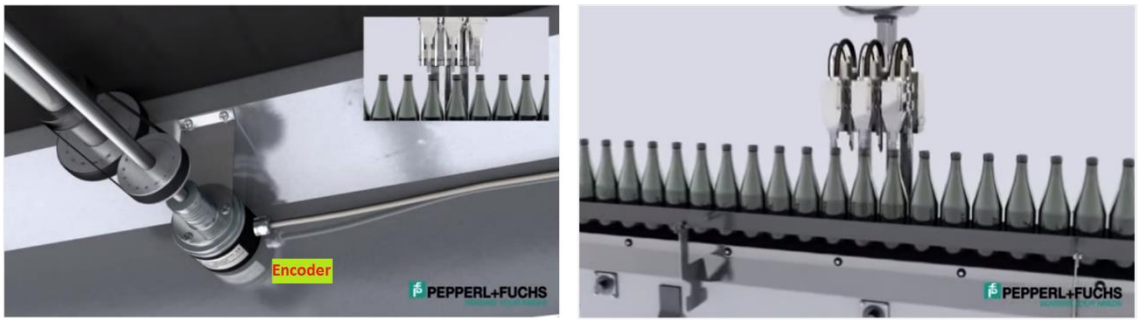


ILUSTRACIÓN 117: DEMO. DOSIFICACIÓN DE BOTELLAS SEGÚN POSICIÓN CONTROLADA POR ENCODER. © PEPPER+FUCHS

Tal y como se puede ver la última ilustración, el encoder, se acopla al eje el cual se quiere obtener su posición, para obtener esta existen dos tipos de codificadores según su construcción mecánica.

Existen dos tipos de codificadores. Absoluto e incremental.

Según [WIK07], el codificador absoluto es definido según su característica mecánica de mantener el control de posición en todo momento.:

An absolute encoder maintains position information when power is removed from the system. The position of the encoder is available immediately on applying power. The

relationship between the encoder value and the physical position of the controlled machinery is set at assembly; the system does not need to return to a calibration point to maintain position accuracy.

Es decir, el codificador mantiene la posición de rotación aun perdiendo la alimentación eléctrica dado que la relación entre el valor del codificador y la posición física de la maquinaria controlada se fija en el ensamblaje y no es necesario que sea recalibrado.

En cambio, según el mismo autor, sobre el codificador incremental indica:

An "incremental" encoder accurately records changes in position, but does not power up with a fixed relation between encoder state and physical position. Devices controlled by incremental encoders may have to "go home" to a fixed reference point to initialize the position measurement.

En este caso, el codificador no posee una relación entre el estado del codificador y la posición física, aunque tiene una alta precisión al cambio de posición de un eje.

¿Cómo se conoce una posición absoluta?

El encoder absoluto tendrá múltiples anillos de código que contienen una serie de ponderaciones binarias que finalmente, proporcionan una palabra de datos que representa la posición absoluta del codificador dentro de una vuelta (360º).

Un disco de metal con una serie de contactos o sensores fotoeléctricos separados del eje principal del es la base de medición en el codificador absoluto, los contactos están posicionados de tal manera que la rotación del codificador ofrezca una combinación binaria dependiendo de los n contactos que estén incluidos en la construcción:

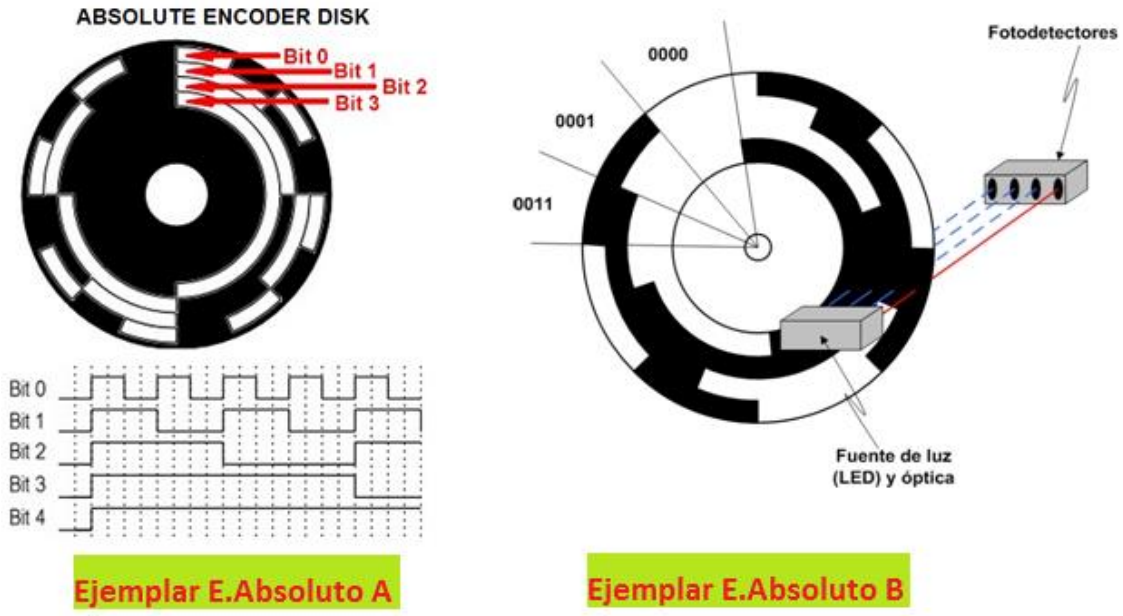


ILUSTRACIÓN 118: DISCO ENCODER ABSOLUTO

Según [WIK07] con n contactos, el número posiciones distintas del es de 2n. Por lo tanto, si existen 3 contactos las posiciones aproximadas serán:

45º a 90º	0	0	1
90º a 135º	0	1	0
135º a 180º	0	1	1
180º a 225º	1	0	0
225º a 270º	1	0	1
270º a 315º	1	1	0
315º 360º	1	1	1

En otro sentido, los codificadores incrementales trabajan de otra forma, el codificador, proporciona salida cíclica solo cuando está girando. Está compuesto por un disco al igual que el absoluto, pero en este caso, está

formado por ranuras, donde dos sensores fotosensibles instalados en el dispositivo permiten detectar la posición del disco.

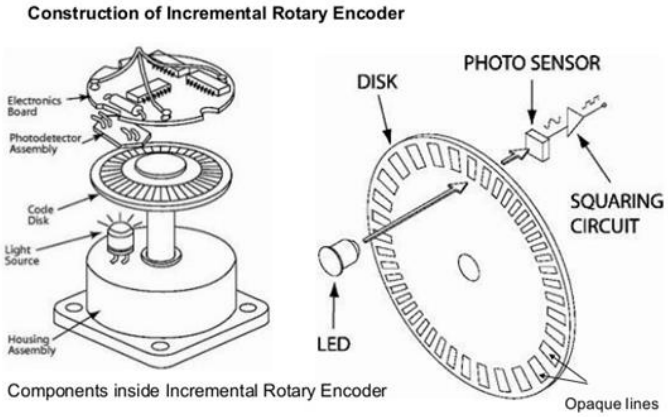


ILUSTRACIÓN 119: CONSTRUCCIÓN DE UN ENCODER INCREMENTAL. © OMRON

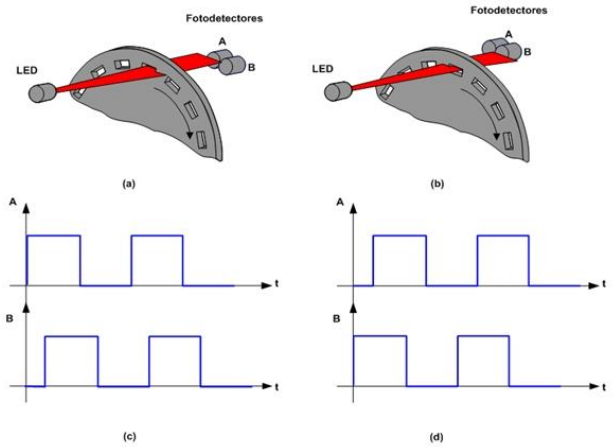


ILUSTRACIÓN 120: POSICIÓN BINARIA SEGÚN LA MEDICIÓN CON DOS FOTOCÉLULAS

Según se puede ver en las imágenes el detector óptico genera un flanco eléctrico cada vez que una de las líneas pasa a través de su campo visual. Posteriormente un controlador cuenta los pulsos para determinar su número por vueltas y tener en cuenta un giro de 360º.

Para mayor precisión, los autores indican que para este codificador puede tener una tercera salida opcional llamada índice de referencia, que ocurre una vez cada vuelta. Esto se utiliza cuando existe la necesidad de una referencia

absoluta, como los sistemas de posicionamiento. La salida del índice suele estar marcada como Z.

Un caso de ejemplo es la posición en un movimiento de precisión de un robot:

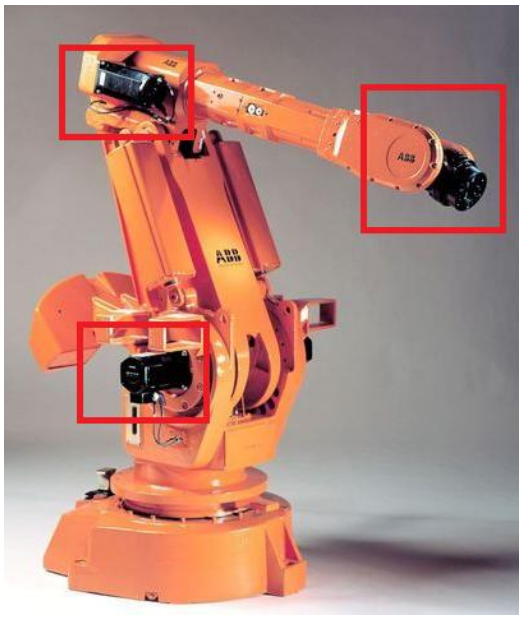


ILUSTRACIÓN 121: SERVOMOTORES DE ROBOT © ABB.

En este caso los propios servomotores que accionan los ejes del robot contienen en su construcción compactado el encoder (en azul):

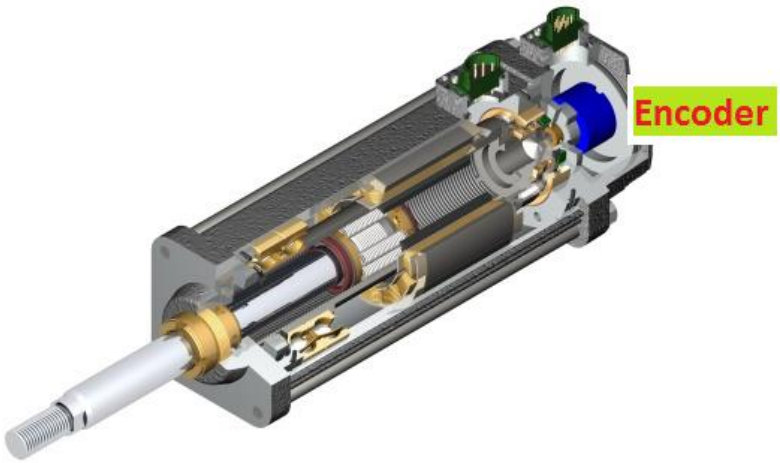


ILUSTRACIÓN 122: ENCODER COMPACTADO EN SERVOMOTOR

En cuanto a los sensores sin tener un contacto físico se habían visto un número mayor de tipos:

ILUSTRACIÓN 123: DETECCIÓN SIN CONTACTO FÍSICO

DETECTORES INDUCTIVOS

El sensor inductivo tiene la capacidad de detectar metales sin tener un contacto físico a partir del principio de las corrientes Foucault [FOU01].

Según [WIK08] y [KEY01]

Los sensores inductivos son una clase especial de sensores que sirve para detectar materiales ferrosos. Son de gran utilización en la industria, tanto para aplicaciones de posicionamiento como para detectar la presencia o ausencia de objetos metálicos en un determinado contexto: detección de paso, de atasco, de codificación y de conteo.

Según el principio de inducción cuando se acerca un metal se genera una inducción magnética que hace que se efectúe un paso de corriente magnética. La amplitud de campo generado será mayor o menor dependiendo de la construcción del imán y el número de espiras del devanado.

La detección tendrá dos estados posibles:

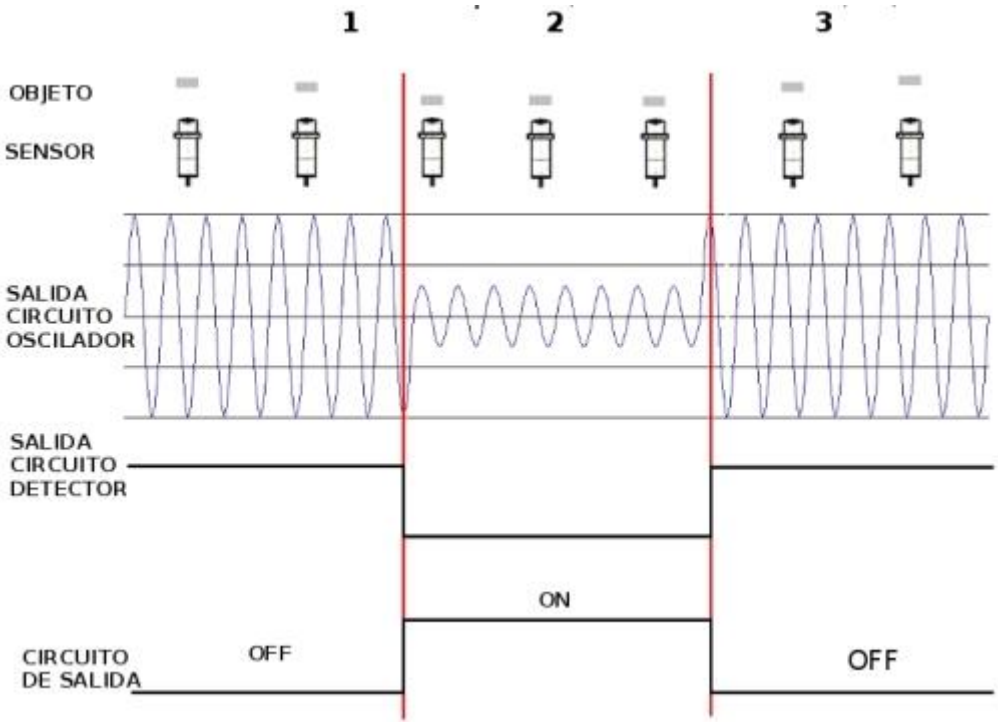


ILUSTRACIÓN 124: ESTADOS DE DETECCIÓN

Un transductor, medirá la inducción magnética generada y a través de un umbral dará una señal de salida ON que será un nivel de tensión.

Para ello el detector inductivo normalmente tiene tres hilos y está alimentado por corriente continua, aunque existen casos que la maniobra donde está conectado es de corriente alterna.

En el caso que la alimentación del sensor sea de 12 Voltios cuando se detecte el metal pasará un flanco con una tensión (Normalmente próxima a los 12 Voltios), que traducido a un circuito lógico digital equivaldrá a un 1 (On) en caso contrario 0 (Off):

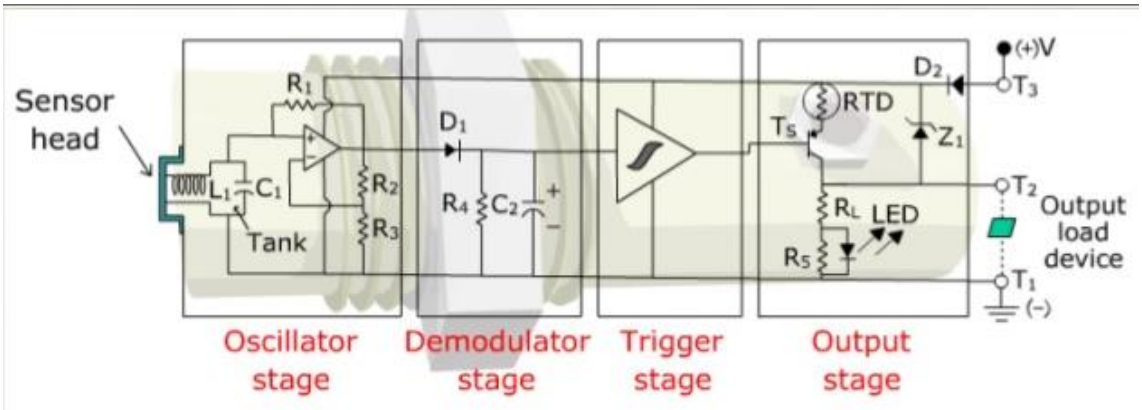


ILUSTRACIÓN 125: FASES DE UN PROCESO DE DETECCIÓN INDUCTIVA

Según el gráfico, el sensor inductivo está construido sobre un proceso de cuatro fases: Oscilador (Donde se produce la inducción magnética), Demodulador donde se trocea la señal, trigger (Que disparará la señal de salida) y la fase de Output, donde se cumple lo indicado en el párrafo anterior, una carga de 12 de Voltios con señal 1 y una carga de tensión la cual generará una diferencia de potencial.

En el caso que este detector sea una entrada a un Autómata programable este conmutará la señal a través de una entrada de este para que emita paso y por lo tanto active una marca o bit interno asociado al input donde está conectado el detector.

Cabe indicar que en esta última fase de salida hay dos tecnologías posibles de transistor. Una referenciada a positivo (PNP) y otra referenciada a polo negativo (NPN), en el proceso referenciado anteriormente se trataría de un detector PNP.

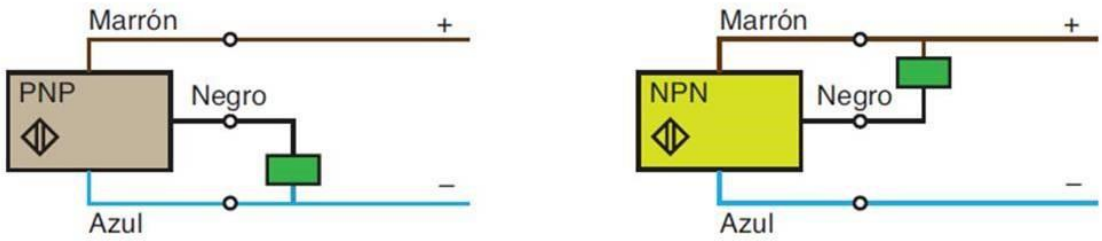


ILUSTRACIÓN 126: DETECTOR PNP Y NPN

Una aplicación en el mundo del motor sería detección de posición y velocidad a partir de la detección de flancos de coronas dentadas, tanto para motores de industriales como los de inducción de corriente alterna como en el cigüeñal de un motor Diesel o gasolina de vehículos de tracción mecánica.

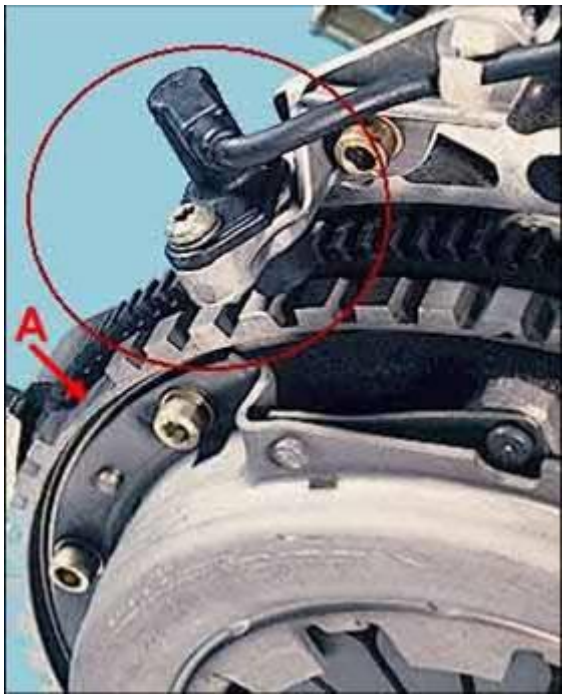


ILUSTRACIÓN 127: DETECCIÓN VELOCIDAD SEGÚN CORONA DENTADA EJE MOTOR [MEC01]

DETECTORES CON EFECTO HALL

Este tipo de sensor sigue el principio Hall el cual, al generar una corriente eléctrica a lo largo de un semiconductor con la proximidad de un campo magnético, se crea una tensión que es perpendicular a la corriente y al campo magnético

Según [WIK09]:

El sensor de efecto Hall o simplemente sensor Hall o sonda Hall (denominado según Edwin Herbert Hall) se sirve del efecto Hall para la medición de campos magnéticos o corrientes o para la determinación de la posición en la que está.

Si fluye corriente por un sensor Hall y se aproxima a un campo magnético que fluye en dirección vertical al sensor, entonces el sensor crea un voltaje saliente proporcional al producto de la fuerza del campo magnético y de la corriente. Si se conoce el valor de la corriente, entonces se puede calcular la fuerza del campo magnético; si se crea el campo magnético por medio de corriente que circula por una bobina o un conductor, entonces se puede medir el valor de la corriente en el conductor o bobina.

Según los autores, estos sensores tienen múltiples funciones entre las cuales destacan:

- Sensores de posición mediante efecto hall operando en modo conmutación tal y como trabaja el sensor inductivo
- Transformadores de corriente como detectores de corriente.
- Indicadores de nivel.
- Medición de espesor de materiales.
- Posición de árboles de transmisión de motores en la industria.
- Posición de cadenas y cigüeñal en vehículos de tracción.

Tal y como se puede observar, existen muchas aplicaciones a este tipo de sensor. Sobre todo, la base es poder tener un elemento que detecta cuerpos magnetizados para emitir una señal de detección.

[LLA01] muestra la utilización de este sensor en el microcontrolador opensource Arduino [ARD01] como poder detectar campos magnéticos, en este monográfico ser presenta el modelo de sensor hall A3144:

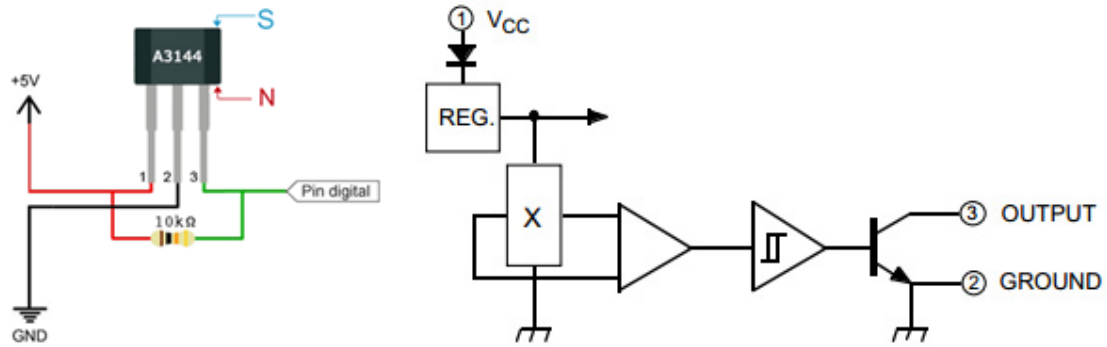


ILUSTRACIÓN 128: BLOQUE FUNCIONAL INTEGRADO A3144 SENSOR HALL

Ante la aproximación de un elemento magnético, el efecto hall generará una diferencia de potencial que emitirá una salida al nivel de tensión del dispositivo Arduino (5 voltios).

DETECTORES CAPACITIVOS

Los detectores capacitivos a diferencia de los inductivos pueden detectar todo tipo de material, aunque no sea conductor tal como el plástico.

Según [SEN01]:

Los detectores capacitivos son "interruptores electrónicos" de característica estática que actúan sin elementos electromecánicos. Su funcionamiento se basa en un circuito oscilante RC y las líneas del campo eléctrico que se cierran a través del aire. La aproximación de un objeto con una constante dieléctrica superior a la del aire, ocasiona el desequilibrio del circuito y el inicio de las oscilaciones.

El autor indica que son sensibles para la mayoría de líquidos y materiales permitiendo detectar presencia de fluidos a través de paredes no conductoras. De la misma forma, afirma que en general los líquidos y sólidos conductores son detectados a una distancia mayor que los materiales aislantes.

Entre las aplicaciones principales se encuentran:

- Detección de nivel.
- Sensor de humedad.
- Detección de posición

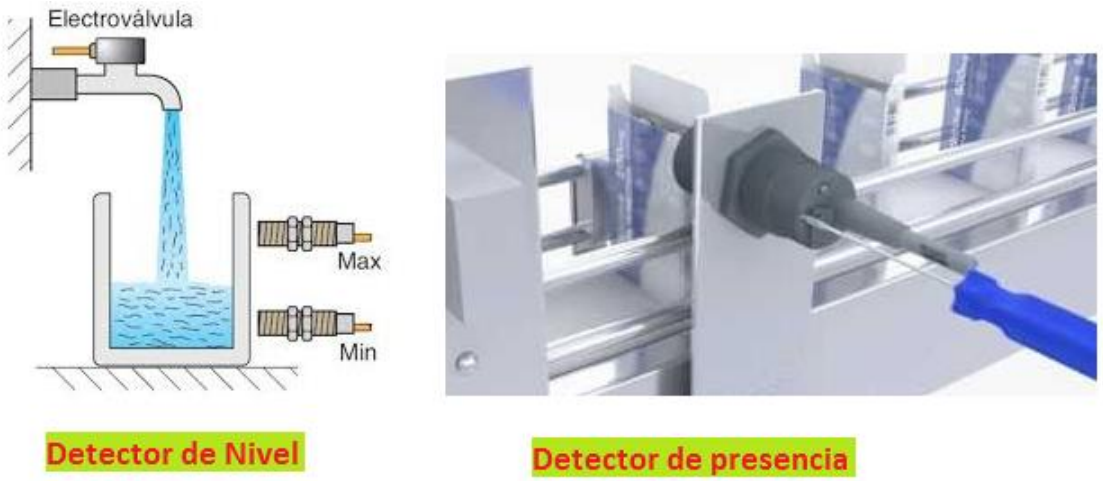


ILUSTRACIÓN 129: DETECCIONES CON DETECTOR CAPACITIVO

En [UVI01] explica el principio de funcionamiento de este tipo de sensor y las distintas aplicaciones en la industria.

Con relación a los motores en la industria el detector capacitivo puede ser aplicado en casos como para detectar presencia de humedad y posición de alguna cinta no metálica cuya tracción se efectúa desde un motor.

DETECTORES ULTRASÓNICOS

Según keyence en [KEY02]:

los sensores ultrasónicos miden la distancia mediante el uso de ondas ultrasónicas. El cabezal emite una onda ultrasónica y recibe la onda reflejada que retorna desde el objeto. Los sensores ultrasónicos miden la distancia al objeto contando el tiempo entre la emisión y la recepción.

En contraste con las células fotoeléctricas, estás tienen un transmisor y receptor mientras que el sensor ultrasónico utiliza un elemento único para la emisión y recepción tal y como muestra la siguiente imagen.

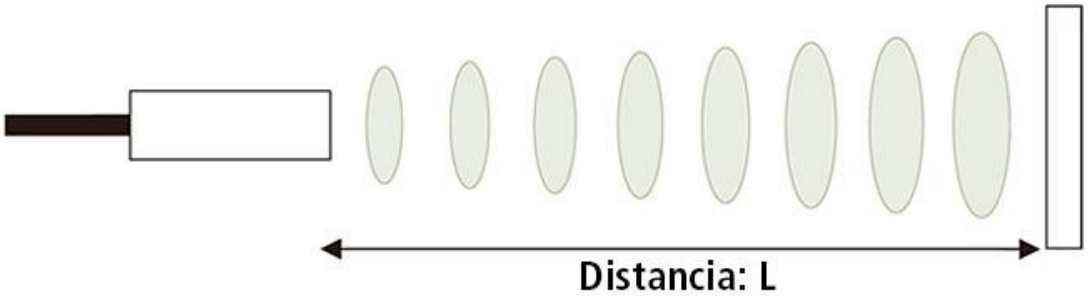


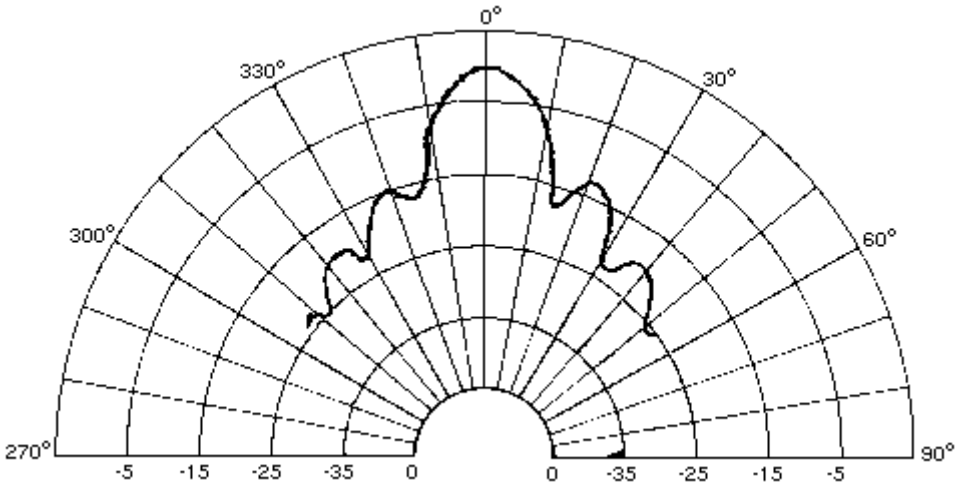
ILUSTRACIÓN 130: SENSOR ULTRASÓNICO. ELEMENTO ÚNICO

No obstante, en [ROB01] indican que existe también el tipo de detector con emisor y receptor separados:

ILUSTRACIÓN 131: MODELO ULTRASÓNICO CON EMISOR Y RECEPTOR SEPARADOS

De la misma forma, los autores muestran las siguientes propiedades de este tipo de sensores:

- *Utilizan cuatro frecuencias de onda: 60Khz, 56Khz, 52.5Khz y 49Khz.*
- *Velocidad del sonido a unos 20ºC es de 343.2m/s.*
- *La onda ultrasónica tiene un ángulo de detección en el cual es teóricamente sensible y por tanto toda onda que venga en un ángulo dentro del lóbulo principal. También existen otros lóbulos secundarios en los cuales el sensor es también sensible, aunque en menor medida. La hoja característica del sensor nos indica la sensibilidad del mismo y la atenuación de la onda (en dBs) según el ángulo de incidencia:*



Según pepperl-fuchs [PEF01], uno de los mayores fabricantes de detectores industriales:

En las aplicaciones industriales, los sensores ultrasónicos se caracterizan por su fiabilidad y excepcional versatilidad. Los sensores ultrasónicos se pueden utilizar para realizar incluso las tareas más complejas relacionadas con la detección de objetos o mediciones de nivel con una precisión milimétrica, ya que su método de medición es fiable en casi todo tipo de condiciones.

En cualquier condición quiere decir que podrá ser utilizado en situaciones de exposición de temperatura o de suciedad que no pueden estar otros dispositivos con menor resistencia.

ILUSTRACIÓN 132: DIFERENTES USOS PARA LA DETECCIÓN POR ULTRASONIDOS. ©PEPPERL-FUCHS

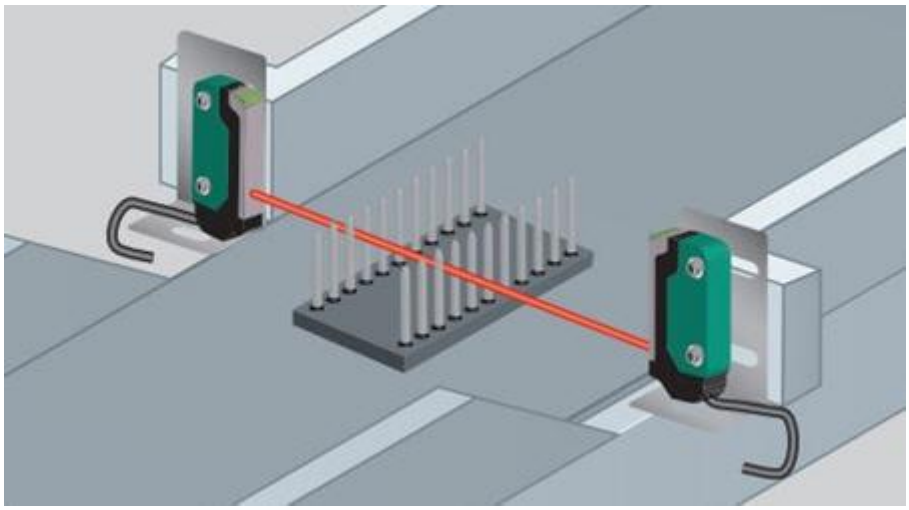
Según [WIK10]:

Un sensor fotoeléctrico o fotocélula es un dispositivo electrónico que responde al cambio en la intensidad de la luz. Estos sensores requieren de un componente emisor que genera la luz, y un componente receptor que percibe la luz generada por el emisor.

Aunque también existen detectores autoreflectantes que únicamente necesita que el haz de luz choque con el objeto a detectar.

Este tipo de sensor es ampliamente usado en la industria sobre cualquier cinta de transporte que necesite detectar elementos que corten el paso del haz de luz, según el fabricante rockwell Allen Bradley [ALB01] Los sensores fotoeléctricos usan un haz de luz para detectar la presencia o la ausencia de un objeto. Siendo esta una alternativa ideal a sensores de proximidad inductivos cuando se requieren distancias de detección largas o cuando el ítem que se desea detectar no es metálico.

El siguiente diagrama en bloques muestra cómo funciona un emisor y un receptor:



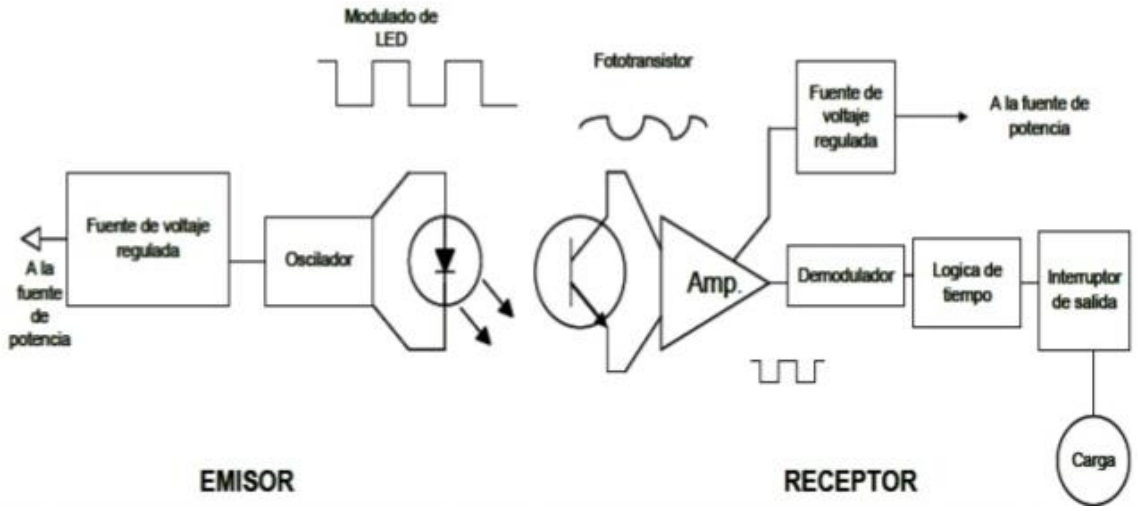


ILUSTRACIÓN 133: BARRERA FOTOELÉCTRICA EMISOR-RECEPTOR

El funcionamiento base se produce cuando un fotodiodo que oscila a una frecuencia emite un haz mientras que la base del fototransistor recibe el haz y amplifica la señal, la cual la demodula a una señal cuadrada y convierte en una carga con un flanco de tensión cada vez que un objeto corta el haz de luz.

Para su correcto funcionamiento el emisor y el receptor deben estar perfectamente alineados para que funcionen los semiconductores fotovoltaicos de emisión y recepción sin el sincronismo entre emisor y receptor no se generará una señal adecuada.

Además de la fabricación por emisión y receptor existen fotocélulas autoreflectantes:

ILUSTRACIÓN 134: FOTOCÉLULA AUTOREFLEXIVA. ©LEUZE

Se caracteriza por tener el emisor en la parte superior y el receptor en la parte inferior.

La última clase es la que se utiliza un reflectante para generar la señal cuando el objeto corta el haz de luz origen.

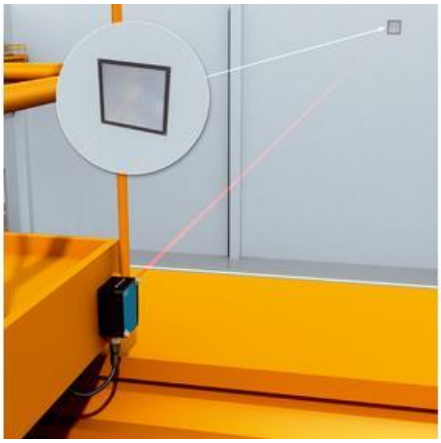


ILUSTRACIÓN 135: CELULA CON ESPEJO REFLECTANTE. ©SICK

Como usos posibles para un motor, la fotocélula puede hacer como regulador de velocidad de una cinta a partir de calcular un buffer de componentes para regular la velocidad. Este caso por ejemplo se usa para evitar cuellos de botella en las líneas de fabricación que están compuestas por diversas estaciones de trabajo separadas por líneas de transporte de producto. Por ejemplo, envases de plástico donde no se puede instalar detectores inductivos para su detección.

En general, este tipo de detección es muy extendida en el campo industrial dado que no está ligado a ningún material para detectar y los fabricantes permiten ajustes para regular la distancia de detección. Esto permite poder adoptar su uso a múltiples tareas de detección. Entre estos a elementos derivados de la tracción de motores como cintas de transporte, cadenas, transmisiones, etc.

DETECCIÓN DE TEMPERATURA

Según [IIA01] se tienen diferentes métodos para medir la temperatura y los categorizan de la siguiente forma:

1. Expansion of a material to give visual indication, pressure, or dimensional change
2. Electrical resistance change
3. Semiconductor characteristic change
4. Voltage generated by dissimilar metals
5. Radiated energy

Es decir, a través de la expansión de un material, cambio en la resistencia eléctrica o características de un semiconductor, el voltaje generado por el calentamiento de dos metales o a través de energía radiada.

A partir de aquí podemos ver la siguiente clasificación de los diferentes medidores de temperatura en relación con el rango de temperatura que son capaces de medir:

- Termocuplas de -200 a 2800 (Diferentes tipos: K,E,J,T,N,B,R,S)
- Sistemas de dilatación (capilares o bimetálicos) de -195 a 760
- Termo resistencias de -250 a 850
- Termistores de -195 a 450
- Pirómetros de radiación de -40 a 4000

Estos sensores los tipificamos de la siguiente forma:

Termocuplas o termopares. Están formados por un bimetal unidos en un extremo, cuando se le aplica una temperatura a esta unión se genera una diferencia de potencial de unos pequeños milivoltios, la relación no es completamente lineal, pero escala al subir la temperatura. Los dos materiales unidos se identifican a partir de una letra característica como la Tipo J (cobre y alumel) o el tipo J (hierro/constantán) el resto se puede ver en [WIK10].

La siguiente gráfica relaciona Temperatura y señal milivoltaica de medición:

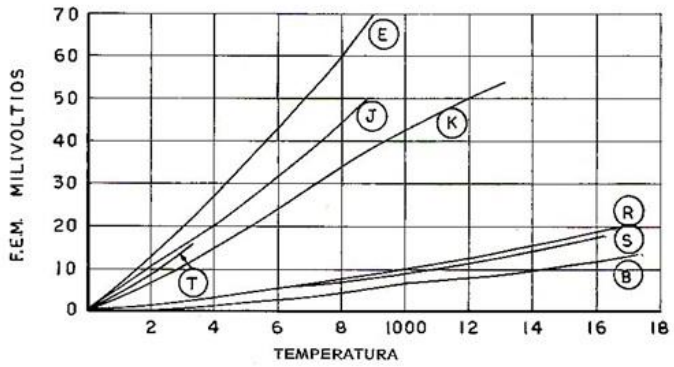


ILUSTRACIÓN 136: RELACIÓN TEMPERATURA - VOLTAJE SEGÚN BIMATERIAL EN UN TERMOPAR

ILUSTRACIÓN 137: TERMOPAR.

Termistores. Está compuesto por una mezcla sintetizada de óxidos metálicos, es básicamente una termo-resistencia, siendo un semiconductor. Existen dos tipos NTC (Negative Temperature Coeficient) y PTC (Positive Temperature coeficient). El efecto es que cuando la temperatura aumenta, para el tipo PTC aumenta la resistencia y para el otro tipo viceversa.

El siguiente gráfico muestra la relación temperatura-resistencia:

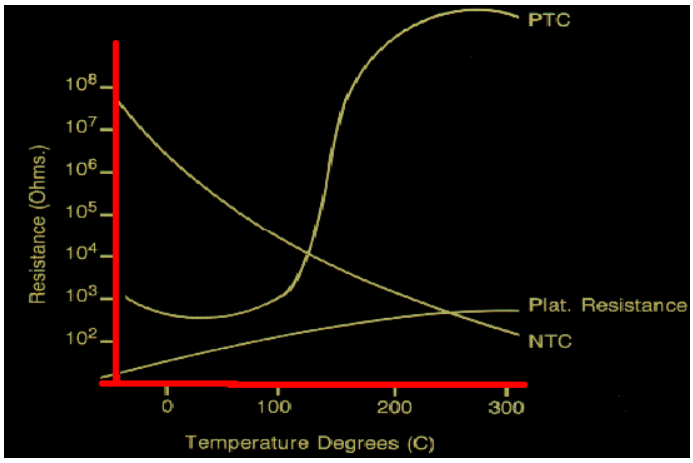


ILUSTRACIÓN 138: RELACIÓN TEMPERATURA-RESISTENCIA EN UNA PTC Y NTC

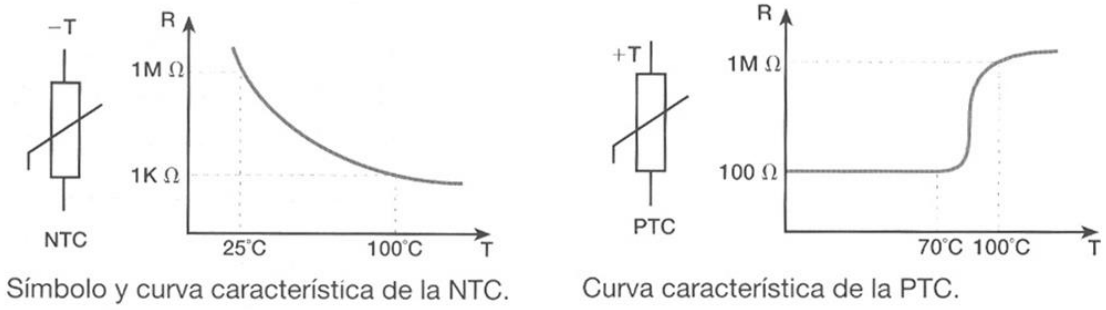


ILUSTRACIÓN 139: CURVAS CARACTERÍSTICAS DE NTC Y PTC

ILUSTRACIÓN 140: SEMICONDUCTORES NTC Y PTC

En general los sensores que varían su resistencia en función de la temperatura son llamados RTD (Resistance Temperature Detector), este tipo de detectores tienen una linealidad entre valor de temperatura y resistencia, dentro de este conjunto de sensores está el PT-100 que su calibración a 0º grados equivale aproximadamente a

100 Ohmios de valor de resistencia. La siguiente tabla muestra la equivalencia de valores resistencia-temperatura:

Temperatura (°C))	0	20	30	40	60	80	100
Resistencia ( )	100	107.79	111.55	115.54	123.1	130.87	138.50

Para la aplicación en motores, los sensores de temperatura pueden ser útiles para poder evaluar directamente el calentamiento de su parte constructiva tanto mecánica (roce del rodamiento, eje calentamiento de carcasa, etc.) como eléctrica (calentamiento de devanados, rotor). También de manera indirecta, midiendo la temperatura a los ejes transmitidos.

RFID (Identificador por radiofrecuencia): Es un sistema que permite recuperar datos y almacenarlos en un dispositivo de forma remota a partir de una tarjeta RFID.

En la literatura de Computación Ubicua ser vio el artículo [BIK01] donde define los componentes principales como:

a) El transportador de RFID, el cual está localizado en un objeto que deber ser identificado y es el soporte de datos
b) El lector RFID, que tiene la capacidad de leer datos y escribirlos en un transmisor.
c) El transmisor puede tratar los datos y/o emitir para emitirlos a otro medio. Puede ser usado con un middleware RFID.

Existen dos tipos de etiquetas. Pasivas y activas.

Las etiquetas pasivas no poseen tensión eléctrica. Reciben una señal de los lectores que induce una diferencia de potencial con una pequeña corriente eléctrica para operar un circuito de tecnología CMOS que está integrado en la etiqueta, estas en su mayoría utilizan basckscatter sobre la portadora que se recibe, una antena lo suficientemente dimensionada tiene la capacidad de dar respuesta. Aunque dada la limitación, estas etiquetas suelen emitir un pequeño código de dato. El coste es mínimo.

Las etiquetas activas, sin embargo, sí poseen alimentación eléctrica, con lo cual puede incluir una electrónica mejor soportada para enviar datos más precisos, el mayor alcance, permite que esta etiqueta pueda estar en ambientes más hostiles como el agua o con temperatura y se pueden operar con ellas a una mejor distancia.

En contraste con otras tecnologías, [BIK01] hace una comparativa con la lectura por infrarrojos, como en la siguiente tabla:

Table 1. : Comparison of Barcode vs. RFID

Attribute	Barcode	RFID
Positive	–Low cost –Broad utilization –Human readable	–No line of sight –Large memory: data moves with product /asset –Dynamic data reads
Negative	–Data transfer requires line of sight –Limited data storage –Environmentally sensitive	–Higher costs –Read sensitive to product attributes –Limited adoption

ILUSTRACIÓN 141: COMPARACIÓN ENTRE CÓDIGO DE BARRAS Y TECNOLOGÍA RFID

Un uso industrial para RFID en contraste con la de código de barras, podría ser la logística de productos, por ejemplo, lectura de productos de limpieza para su distribución o por ejemplo lectura de bastidores de vehículos una vez han salido de fábrica.

Siemens incorpora una gama de productos para realizar lectura:

Descripción
SIMATIC RF200 es el nuevo sistema RFID compacto dentro de la familia de productos SIMATIC RF. Esta línea de productos se compone de lectores HF económicos y especialmente apropiados para aplicaciones del ámbito de las líneas de montaje pequeñas o de la intralogística. Los lectores SIMATIC RF200 RFID asisten exclusivamente el estándar RFID ISO 15693 y por tanto están concebidos para el uso con la completa gama de productos de transpondedores MOBY D. Como parte de nuestro portfolio completo de sistemas de identificación, SIMATIC RF200 ofrece un acceso compacto para aplicaciones de la gama de prestaciones inferior a media. Para tareas de identificación sencillas se dispone de los nuevos lectores SIMATIC RF210R, RF220R y RF260R con la variante de interfaz para IO-Link. En la nueva versión mejorada los lectores ahora también son capaces de leer y escribir en el entorno IO-Link.

Sistema RFID	SIMATIC RF200
Distancia de escritura/lectura	hasta 130 mm
Frecuencia	13,56 MHz
Estándares	ISO 15693
Variantes de interfaz	RS422 RS232 IO-Link

ILUSTRACIÓN 142: SISTEMA RFID © SIEMENS

En las siguientes líneas se muestran las tablas que intervendrán en el modelo de datos del sistema.

Entidad	REGISTRO_HISTORICO			
Campo	**Tipo de Dato**	**Tamaño**	**Clave**	**Detalle**
Id_historico	Integer	10		
Tipo_historico	Varchar	100		{Cadencia, rechazos, Parada_estacion, parada_transporte, frecuencia_problemas}
Fecha_peticion	Datetime			
Linea_envasado	Varchar	100		
Estacion_trabajo	Varchar	100		

TABLA 6: ENTIDAD BBDD REGISTRO_HISTORICO

Entidad	ESTACION_TRABAJO			
Campo	**Tipo de Dato**	**Tamaño**	**Clave**	**Detalle**
Id_estacion	Integer	10	Primaria	
Linea_envasado	Integer	10		
Fecha_Alta	Date			
Cod_estacion	Varchar	100		
Desc_estación	Varchar	100		
Es_transporte	Boolean			

TABLA 7: ENTIDAD BBDD ESTACION_TRABAJO

Entidad	CADENCIA			
Campo	**Tipo de Dato**	**Tamaño**	**Clave**	**Detalle**
Id_cadencia	Integer	10	Primaria	
envase_hora	Integer	10		
Fecha	Date			
Linea_envasado	Varchar	100		
Estacion_trabajo	Varchar	100		
Transporte	Varchar	100		

TABLA 8: ENTIDAD BBDD CADENCIA

Entidad	PROBLEMA			
Campo	**Tipo de Dato**	**Tamaño**	**Clave**	**Detalle**
Id_problema	Integer	10	Primaria	
TipoProblema	Varchar	100		

Descripcion_Problema	Varchar	300		
Estacion_trabajo	Varchar	100		
Parada	Varchar	100		
Fecha Inicio	Date			
Fecha fin	Date			

TABLA 9: ENTIDAD BBDD PROBLEMA

Entidad	TIPO_PROBLEMA			
Campo	Tipo de Dato	Tamaño	Clave	Detalle
Id_tipo_problema	Integer		Primaria	
Cod_problema	Varchar	100		
Desc_problema	Varchar	100		
Fecha	Data	100		

TABLA 10: ENTIDAD BBDD TIPO_PROBLEMA

Entidad	REPORTE_DATOS_ESTACION			
Campo	Tipo de Dato	Tamaño	Clave	Detalle
Id_reporte	Integer	10	Primaria	
Desc_reporte	Varchar	100		
Fecha_hora	Datetime			
Tipo de reporte	Varchar	100		{TURNO, DIA, HORA}
Ruta_filesystem	Varchar	100		

TABLA 11: ENTIDAD BBDD REPORTE_DATOS_ESTACION

Entidad	PARADA			
Campo	Tipo de Dato	Tamaño	Clave	Detalle
Id_ Parada	Integer	10	Primaria	
Tipo_Parada	Varchar	100		{Cambio formato, avería}
Desc_Parada	Varchar	100		
Hora_Inicio	Datetime			
Hora_Fin	Datetime			
Estacion_trabajo	Varchar	100		
Param_num_parada	Varchar	100		{5min, 10min, 15min...}

TABLA 12: ENTIDAD BBDD PARADA

Entidad	CONTROLADOR_IOT			
Campo	Tipo de Dato	Tamaño	Clave	Detalle
Id_controlador	Integer	10	Primaria	
Estación_trabajo	Varchar	100		{SERAC1, SERAC2, ERCA1...}
Elementos_entrada	Varchar	100		
Elementos_salida	Varchar	100		

TABLA 13: ENTIDAD BBDD CONTROLADOR_IOT

Entidad	RFID			
Campo	Tipo de Dato	Tamaño	Clave	Detalle
Codigo_rfid	Integer	10	Primaria	
Linea_envasado	Varchar	100		{SERAC1, SERAC2, ERCA1...}
Tipo_producto	Varchar	100		{A, B, C, D, E, F}
Hora_deteccion	Datetime			

Estacion_trabajo	Varchar	100		
Turno	Varchar	100		{M, T, N}
Id_controlador	Varchar	100		

TABLA 14: ENTIDAD BBDD RFID

Entidad	LOTE			
Campo	Tipo de Dato	Tamaño	Clave	Detalle
Cod_lote	Integer	10	Primaria	
Dia_Hora_inicio	Datetime			
Dia_Hora_Fin	Datetime			

TABLA 15: ENTIDAD BBDD LOTE

Entidad	CAMBIO_FORMATO			
Campo	Tipo de Dato	Tamaño	Clave	Detalle
Id_Cambio	Integer		Primaria	
Fecha_cambio	Datetime			
Linea_envasado	Varchar	100		
Estacion_trabajo	Varchar	100		
Lote	Varchar	100		
Cod_parada	Integer	10		
Incidencia	Varchar	100		{0,1, 2,}

TABLA 16: ENTIDAD BBDD CAMBIO_FORMATO

Entidad	INCIDENCIA			
Campo	Tipo de Dato	Tamaño	Clave	Detalle
Id_incidencia	integer	10	Primaria	
Cod_incidencia	Varchar	100		{0-Sin incidencia, 1-Prolongación tiempo, 2-...}
Desc_incidencia	Varchar	100		

TABLA 17: ENTIDAD BBDD INCIDENCIA

Entidad	AVERIA			
Campo	Tipo de Dato	Tamaño	Clave	Detalle
Id_averia	Integer	10	Primaria	
Cod_Averia	Varchar	50		
Desc_Averia	Varchar	100		
Linea_envasado	Varchar	100		
Estacion_trabajo	Varchar	100		
Fecha	Datetime			

TABLA 18: ENTIDAD BBDD AVERIA

Entidad	REGISTRO_RECHAZOS			
Campo	Tipo de Dato	Tamaño	Clave	Detalle
Id_registro	Integer	10	Primaria	
Fecha	Datetime	100		
Linea_envasado	Varchar	100		
Estacion_trabajo	Varchar	100		
Num_Rechazos	Integer	10		

Turno	Varchar	100		
Lote	Varchar	100		

TABLA 19: ENTIDAD BBDD REGISTRO_RECHAZOS

Entidad	DATOS_CADENCIA_DEPENDENCIAS			
Campo	Tipo de Dato	Tamaño	Clave	Detalle
Id_registro_dependencias	Integer	10	Primaria	
Fecha	Varchar	100		
Estacion_trabajo	Varchar	100		
Es_transporte	Boolean			
Linea_envasado	Varchar	100		
cadencia	Varchar	100		

TABLA 20: ENTIDAD BBDD DATOS_CADENCIA_DEPENDENCIAS

Entidad	D_SENSOR_INTELIGENTE_IN			
Campo	Tipo de Dato	Tamaño	Clave	Detalle
Id_sensor_inteligente	Integer	10	Primaria	
Frecuencia_envase	Varchar	100		
Tiempo_medio_envase	Varchar	100		
Fecha_Dato	Datetime			
Otro_Dato_clave	Varchar	100		

TABLA 21: ENTIDAD BBDD D_SENSOR_INTELIGENTE_IN

Entidad	USUARIO			
Campo	Tipo de Dato	Tamaño	Clave	Detalle
Id_usuario	Integer	10	Primaria	
Id_persona	Integer	10		
Rol	Integer	10		
Linea_asignada	Varchar	100		
Fecha_alta	Datetime			

TABLA 22: ENTIDAD BBDD USUARIO

Entidad	PERSONA			
Campo	Tipo de Dato	Tamaño	Clave	Detalle
NIF	Varchar	100	Primaria	
Nombre	Varchar	100		
Apellido_1	Varchar	100		
Apellido_2	Varchar	100		
Domicilio	Varchar	100		
Factoria	Varchar	100		

TABLA 23: ENTIDAD BBDD PERSONA

Entidad	ROL			
Campo	Tipo de Dato	Tamaño	Clave	Detalle
Id_rol	Integer	10	Primaria	
Cod_rol	Varchar	100		
Desc_rol	Varchar	100		

TABLA 24: ENTIDAD BBDD ROL

En primer lugar, se muestra la secuencia de instancias y comunicación entre objetos a alto nivel que realizará un usuario al acceder a través del navegador para monitorizar la estación.

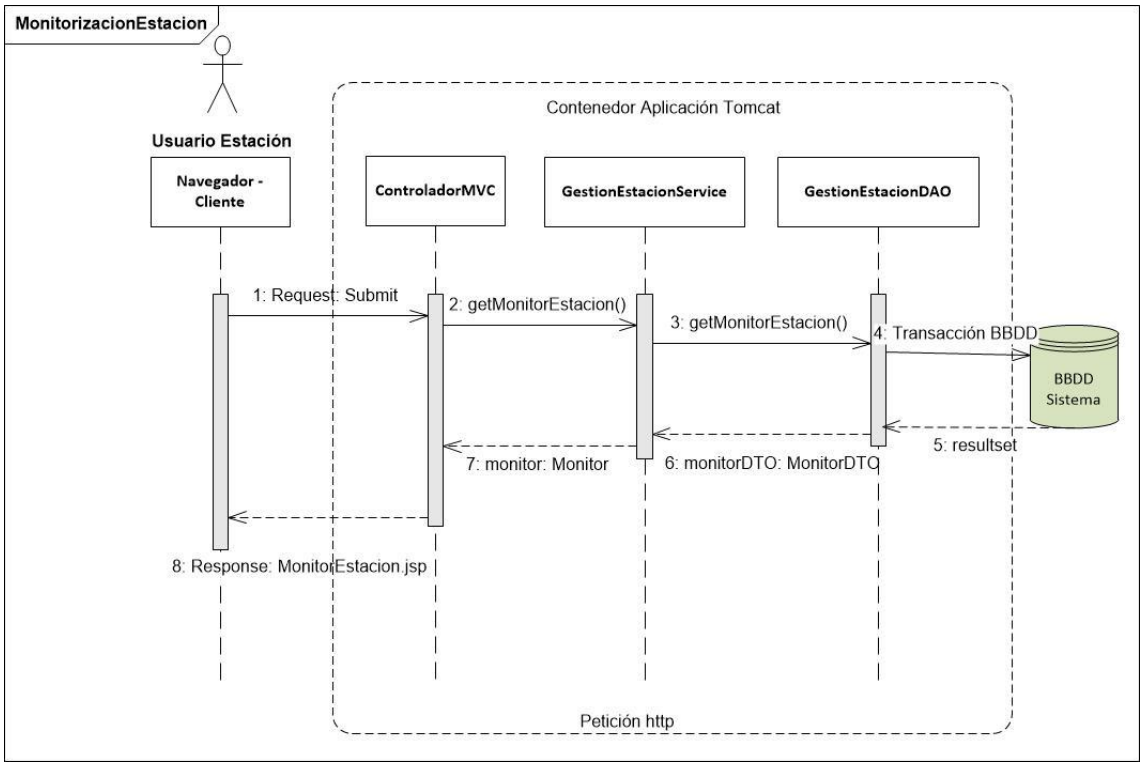


ILUSTRACIÓN 143: DIAGRAMA DE SECUENCIA PRINCIPAL FUNCIONALIDAD MONITORIZACIÓN ESTACIÓN

El siguiente diagrama de secuencia relata los pasos que se deberán de realizar dentro de la funcionalidad de gestión de reporte online la selección de rechazos en una fecha determinada:

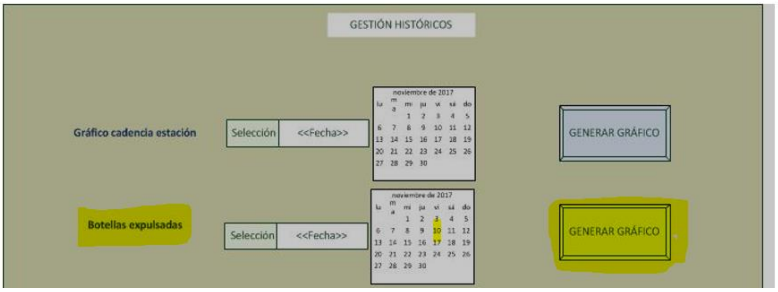


ILUSTRACIÓN 144: RECORTE MAQUETA GESTIÓN HISTÓRICOS, TAREA DE GENERAR GRÁFICO BOTELLAS RECHAZADAS

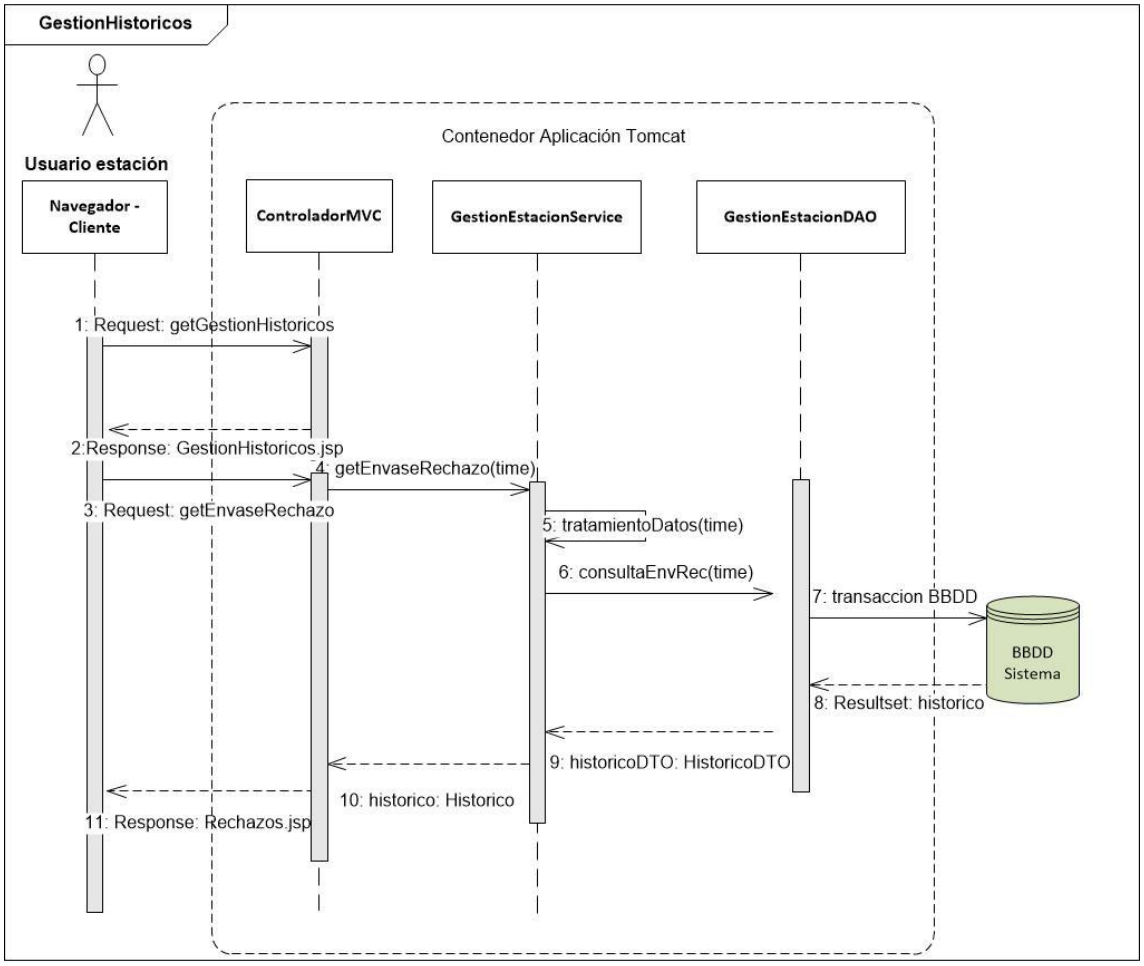


ILUSTRACIÓN 145: DIAGRAMA DE SECUENCIA PRINCIPAL DE FUNCIONALIDAD DE GESTIÓN DE HISTÓRICOS

El siguiente diagrama modela la gestión de un reporte tipo informe diario:

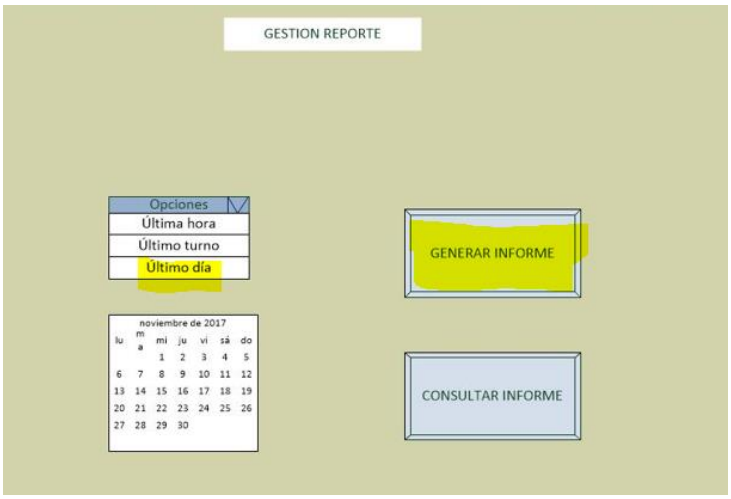


ILUSTRACIÓN 146: RECORTE MAQUETA GESTIÓN REPORTE. TAREA DE GENERAR INFORME DEL DIA

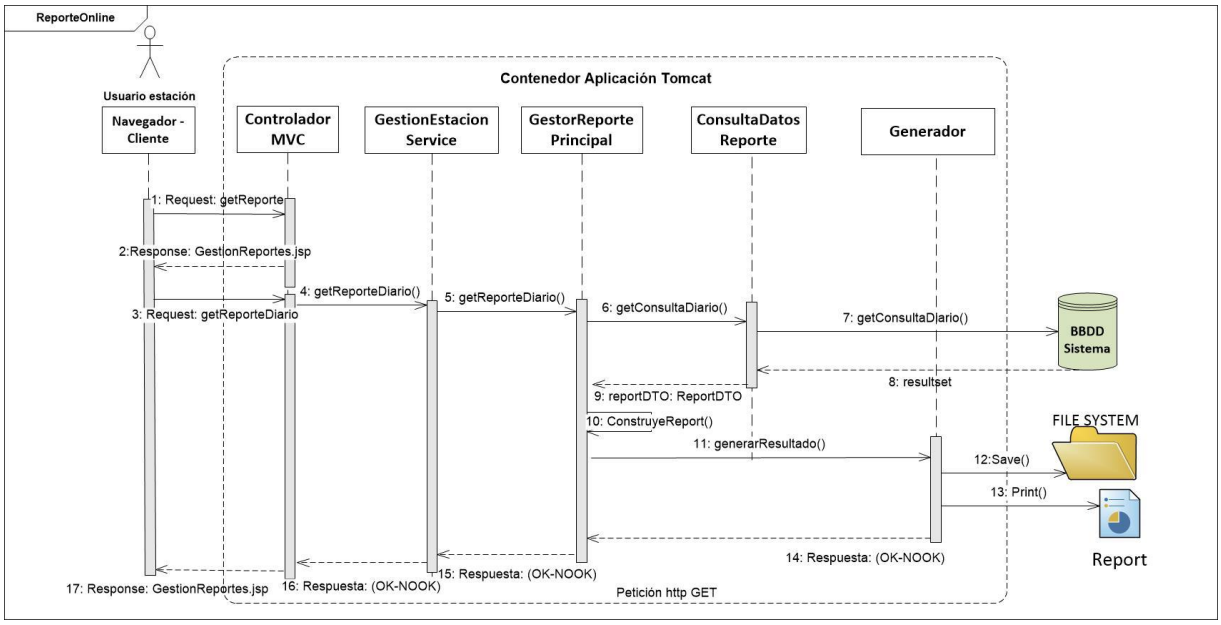


ILUSTRACIÓN 147: DIAGRAMA DE SECUENCIA PRINCIPAL TAREA GENERAR INFORME DEL DIA

En cuanto al proceso generado por la detección de los sensores del controlador IoT, los siguientes diagramas de secuencia muestran la secuencia desde el procesado de la señal del sensor hasta su persistencia:

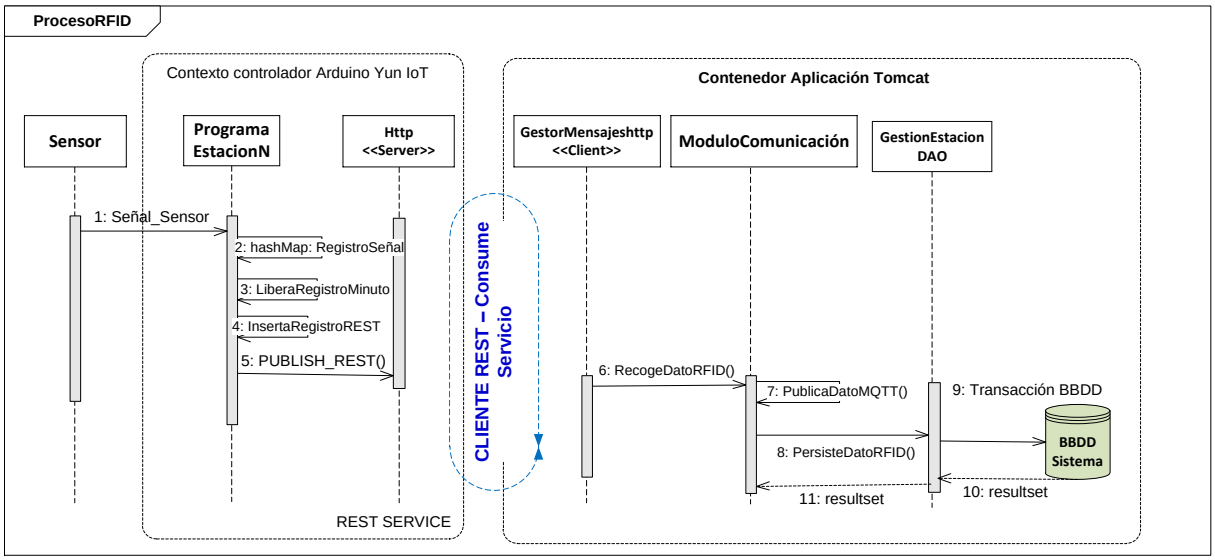


ILUSTRACIÓN 148: DIAGRAMA DE SECUENCIA PROCESO RFID

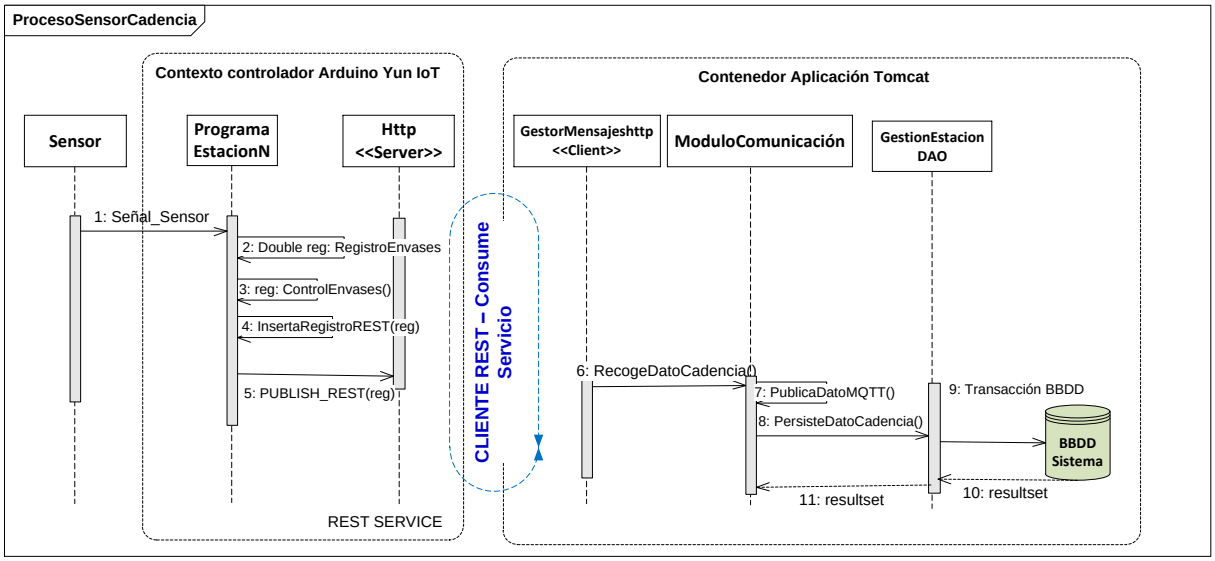


ILUSTRACIÓN 149: DIAGRAMA DE SECUENCIA PROCESO DE SENSOR CADENCIA

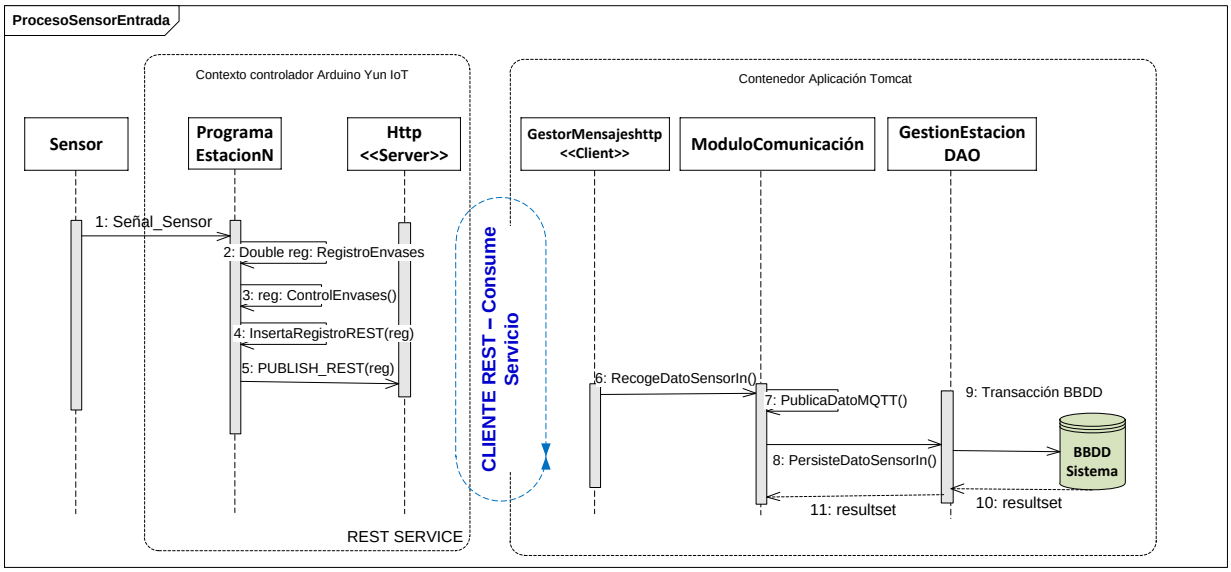


ILUSTRACIÓN 150: DIAGRAMA DE SECUENCIA TRATAMIENTO SENSOR ENTRADA

En la siguiente línea se muestra un diagrama de secuencia a alto nivel de un modelo de secuencia de envío a partir del protocolo TCP del controlador externo a las dependencias del presente sistema:

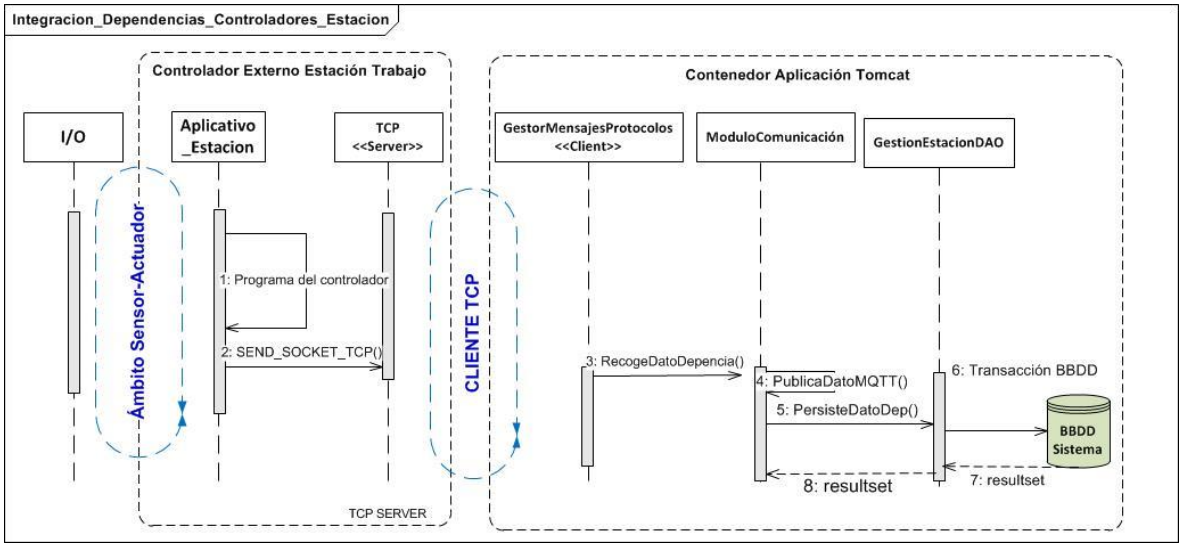


ILUSTRACIÓN 151: DIAGRAMA DE SECUENCIA TRATAMIENTO DE DATO TCP DE DEPENDENCIA EXTERNA