

DETECCIÓN DE FALLAS EN TIEMPO REAL MEDIANTE REDES COMPLEJAS EN UN SISTEMA DE MANUFACTURA 4.0

FAILURE DETECTION IN REAL TIME THROUGH COMPLEX NETWORKS IN A 4.0 MANUFACTURE SYSTEM

Josué Antonio Prieto Olivares

Tecnológico Nacional de México / IT de Aguascalientes, México
G19150597@aguascalientes.tecnm.mx

Elvia Ruiz Beltrán

Tecnológico Nacional de México / IT de Aguascalientes, México
eruiz@aguascalientes.tecnm.mx

Jorge Luis Orozco Mora

Tecnológico Nacional de México / IT de Aguascalientes, México
Jorge.om@aguascalientes.tecnm.mx

Jorge Octavio Valdés Valadez

Tecnológico Nacional de México / IT de Aguascalientes, México
Jorge.vv@aguascalientes.tecnm.mx

Recepción: 28/octubre/2020

Aceptación: 2/diciembre/2020

Resumen

En el entorno de la industria 4.0 todos los elementos están conectados entre sí, es por ello por lo que, para la detección de fallas, actualmente se agregan más sensores al proceso en algunas etapas, esta actividad se realiza para tener un monitoreo completo del sistema, dicha solución puede ser costosa. A través de la metodología de modelado estructural por redes complejas, se puede representar todas las características dinámicas del sistema de manufactura a evaluar. Con información de los tiempos de procesos, se presenta una técnica de búsqueda la cual puede detectar las fallas del sistema cuando este no evoluciona correctamente a la etapa siguiente del proceso. Al utilizar como información mínima los tiempos de evolución, se elimina completamente el uso de sensores extras al proceso ya existente.

Palabras Clave: Detección de fallas, Factory IO, Industria 4.0, LabVIEW, Redes complejas.

Abstract

In the industry 4.0 environment, all elements are connected to each other, that is why, for failure detection, currently sensors are added to the process in some stages, this activity is carried out to have a complete monitoring of the system, this solution can be expensive. Through the structural modeling methodology for complex networks, all the dynamic characteristics of the manufacturing system to be evaluated can be represented. With information about the process time, an algorithm can detect system failures when it does not evolve correctly to the next stage of the process. By using evolution time as minimum information, the use of extra sensors to the existing process is completely eliminated.

Keywords: *Complex Networks, Factory IO, Failure Detection, Industry 4.0, LabVIEW.*

1. Introducción

En este trabajo, se propone una metodología de detección de fallas basada la teoría de redes compleja con la cual se modeló un sistema de manufactura. Cabe destacar que el caso de estudio del artículo se puede extender a un proceso industrial mil o más elementos interactuando entre sí. Lo cual muestra el potencial del uso de esta teoría.

En la actualidad los sistemas de manufactura 4.0 comienzan a ser de suma importancia ya que se piensa en empresas inteligentes con un alto grado autonomía en el intercambio de información y en el control de procesos [Tjahjono, 2017]. En los últimos años, la humanidad se ha rodeado de sistemas que tienden a tener muchos componentes que interactúan entre sí [Reyes, 2015], lo cual se vuelve complicado para la predicción en su funcionamiento y por tal motivo tiene alta posibilidad a fallar. A estos sistemas se les conoce como sistemas complejos [Barabási, 2015]. En general los sistemas de uso continuo tienden a fallar de manera aleatoria y desconocida para los usuarios, es por este motivo que el detectar las

fallas en un intervalo de tiempo suficientemente corto puede resultar un ahorro en tiempo y dinero para la empresa, además de reducir la probabilidad de tener equipos dañados. [García, 2015].

En el presente trabajo a diferencia de [Preciado, 2015] no se añadieron sensores extras para hacer la identificación de una falla, al contrario, se logra trabajar con el sistema original sin modificaciones. La metodología mostrada en este artículo tiene sustento en la forma de modelar estructuralmente redes complejas mostrada por [Bonacina, 2019], [Preciado, 2015] y [Kleinberg, 2017], en donde la conectividad entre los nodos y los links son elementos de los sistemas físicos, tales como actuadores y transductores, es decir, PLCs, motores, electroválvulas, cilindros neumáticos, sensores, entre otros.

Un sistema complejo, se define como aquel que está compuesto por una cantidad enorme de partes que interactúan entre sí, de tal manera que no es trivial determinar el efecto de una entrada a las salidas del sistema. Cada parte tiene su propia estructura y cumple una función específica, esto quiere decir que lo que ocurra con cierta parte del sistema, influirá a todo éste. Si ocurre una falla su efecto se propagará sobre el resto de los elementos del sistema, a este tipo de fallas se le llama, fallas en cascada. En este tipo de sistemas se presentan comportamientos dinámicos continuos y de estados discretos que evolucionan impulsados por eventos [Preciado, 2015]. El análisis de fallas en sistemas complejos ha sido recientemente abordado para analizar sistemas de manufactura flexible y fallas en cascada en redes de distribución eléctrica; dichas metodologías aplicadas resultan ser exitosas ya que permiten detectar y localizar fallas en los sistemas de manera precisa, sin embargo, la forma de realizar el monitoreo es mediante la integración de observadores de estados (sensores extras al proceso) dicha inventiva resulta ser costosa y complicada de acoplar a la instalación actual.

Bajo una metodología similar de detección a la que utiliza [Dhal, 2015], se identifica la falla en los links de la red compleja, sólo que a diferencia del artículo antes mencionado, tienen una medición en cada nodo, esto es conveniente para el tipo de problema que se está planteando en este artículo ya que consta de pocos nodos (estados) y sin problema se podría colocar un observador en cada nodo pero la

razón de ser de esta investigación es, el poder migrar la técnica a sistemas realmente complejos con más de mil nodos, mismos que tienen miles de elementos conectados entre sí, los cuales se vuelven difícil de implementar cuando la técnica de detección involucra la instalación de elementos de medición (sensores).

Actualmente con la cuarta revolución industrial (Industria 4.0) que es una nueva tendencia de automatización, gracias a las nuevas tecnologías de internet de las cosas (IoT) se puede hacer uso de elementos de monitoreo remoto y de modelados estructurales de control y optimización [Mastos, 2020]. Dadas las consultas de información previas se conocen que existen modelado de sistemas de manufactura con redes complejas capaz de hacer la discriminación de una falla aleatoria en cualquier instante de tiempo [Preciado, 2015], [Reyes, 2015], [Shang, 2016] y [Zhao, 2017]. Lo cual logra ser aplicable para algunos casos particulares, con pocos elementos o nodos. En este documento se externa una solución a sistemas realmente complejos de más de mil elementos interconectados entre sí, con la información mínima de sólo conocer sus propiedades estructurales y los tiempos de evolución del sistema.

2. Métodos

En el presente trabajo se realiza una aplicación, a mínima escala con menos de 20 elementos (Motores, sensores, botones, etc.), de lo que se puede lograr con esta metodología de detección de fallas basada de redes complejas en tiempo real con tecnología IoT, es importante recalcar que lo mostrado aquí es sólo una demostración con un sistema de manufactura con menos de mil elementos, pero la flexibilidad que da la representación en redes complejas puede implementarse en sistemas de más de mil elementos interconectados entre sí.

Identificar las características particulares del sistema de manufactura

Con ayuda del software Factory IO [Real Games, 2020] se diseñó el siguiente caso de estudio/sistema de manufactura, ver figura 1.

El funcionamiento del sistema consiste en clasificar cajas de acuerdo con su peso, dicho sistema puede separar tres tipos de cajas, pequeña, mediana y grande. Todo

comienza al presionar el botón de arranque (BA), el siguiente evento es la creación de una caja aleatoria a partir del alimentador (Emitter) concentrado en la banda transportadora 1 (B1), figura 2. Posteriormente al ser detectado por el sensor uno (At scale entry), éste activa la siguiente banda (B2) donde al censarse en posición (At scale), es pesado por el sensor analógico de peso (Sap), figura 3.



Figura 1 Sistema de manufactura 4.0 propuesto.

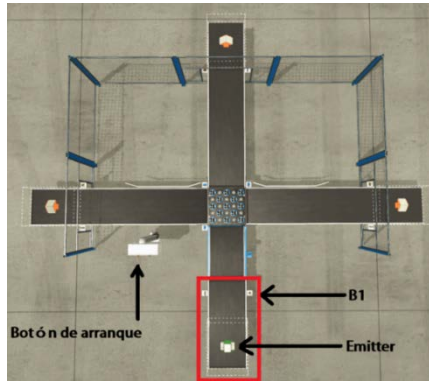


Figura 2 Parte principal del proceso.

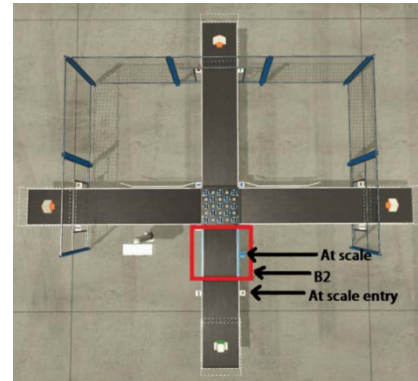


Figura 3 Zona de pesaje para cajas.

En el centro del proceso se encuentra un selector quien puede moverse a la izquierda, derecha y frontal (MSI, MSD, MSF), donde se toma la decisión de cual lado es el favorable para el peso de la caja, las pequeñas se van por el lado izquierdo, las medianas por el derecho y las grandes por la parte central, figura 4. En el lado izquierdo, derecho y central se observa un sensor en el comienzo de las bandas transportadoras finales (B3, B4, B5 respectivamente), los sensores de cada banda (At left entry, At right entry y at forward entry respectivamente) mandan una señal al controlador para activar las bandas antes mencionadas. Al final de cada

una de las bandas transportadoras se encuentra una zona en donde la caja (pequeña, mediana y grande) termina su proceso, un sensor (At exit left, At exit right y at exit front respectivamente) al detectarlas reinicia nuevamente el sistema para la obtención de una nueva caja, figura 5.

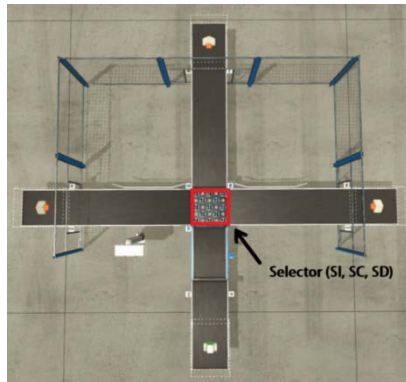


Figura 4 Selector.

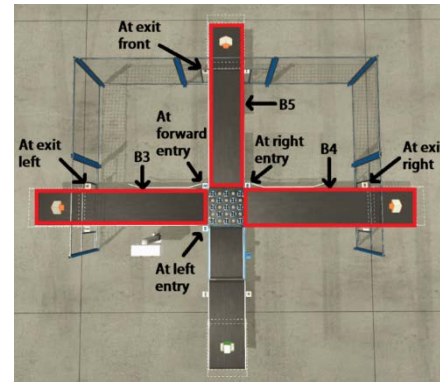


Figura 5 Etapas final del proceso.

El sistema consta en general de entradas y salidas, las cuales son representadas en la figura 6. En la tabla 1 se describe la relación de los elementos de entrada del sistema de manufactura 4.0.

Del mismo modo el sistema consta de actuadores o salidas, las cuales están representadas en la figura 7. En la tabla 2 se agrupan los elementos de salida al igual que sus características.

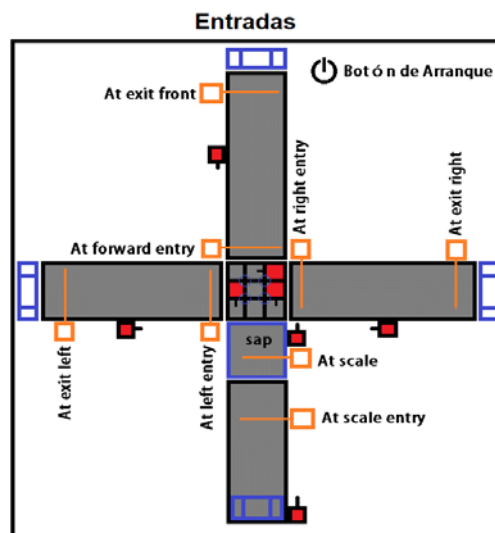


Figura 6 Etiquetado de entradas del sistema.

Tabla 1 Tabla de entradas.

Etiqueta	Descripción	Naturaleza
BA	Botón de arranque de sistema	Digital
At scale entry	Sensor principal	Digital
At Scale	Sensor en posición de pesaje	Digital
Sap	Sensor analógico de peso	Analógico
At left entry	Sensor izquierdo posterior a selector	Digital
At forward entry	Sensor frontal posterior a selector	Digital
At right entry	Sensor derecho posterior a selector	Digital
At exit left	Sensor final izquierdo, caja lista	Digital
At exit front	Sensor final frontal, caja lista	Digital
At exit right	Sensor final derecho, caja lista	Digital

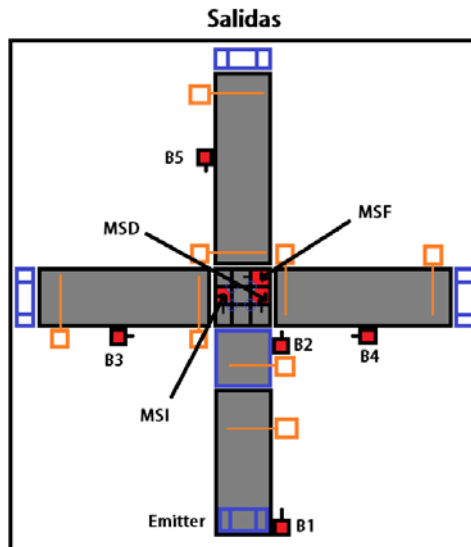


Figura 7 Etiquetado de salidas del sistema.

Tabla 2 Tabla de salidas o actuadores.

Etiqueta	Descripción	Naturaleza
Emitter	Generador aleatorio de cajas	Digital
B1	Motor banda uno	Digital
B2	Motor banda dos	Digital
B3	Motor banda tres	Digital
B4	Motor banda cuatro	Digital
B5	Motor banda cinco	Digital
MSI	Motor selector izquierdo	Digital
MSD	Motor selector derecho	Digital
MSF	Motor selector frontal	Digital

Cabe destacar que las limitantes que integran este análisis son debido al software de simulación Factory IO, en el cual algunos elementos no son observables, esto quiere decir, que no se puede medir el estado de algunos elementos.

Modelado estructural en red compleja

Para realizar una abstracción del sistema de manufactura a analizar, se consta de dos elementos principales, nodos y links, en los cuales para este trabajo los nodos son todos aquellos actuadores o salidas del sistema, por lo tanto, consta de 9 nodos, la conexión de los links representa la relación secuencial que tiene cada elemento, por ejemplo, para iniciar el sistema es necesario presionar el botón de arranque, mismo que representa un link que cambia del estado cero (estado de reposo), al estado uno. El modelo estructural en red compleja del sistema de manufactura de la figura 1 se muestra en la figura 8, misma que integra todos los eventos posibles (nodos) previamente identificados.

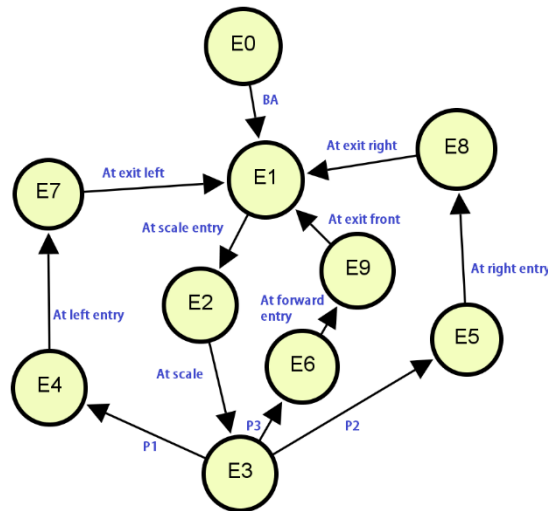


Figura 8 Modelo del sistema de manufactura de la figura 1 en red compleja.

Se decide utilizar las redes complejas como técnica de detección debido a que existe una ventaja muy marcada, y es que no importa la cantidad de elementos inmiscuidos, siempre se tendrá una representación estructural del sistema.

Las etiquetas propuestas en los nodos de la red (figura 8) consisten en los eventos que pueden ser detectables en caso de falla. Mismos que se agrupan en la tabla 3, los links que aparecen son las entradas que previamente ya se mostraban en la tabla 1. Se aprecian los eventos que son detectables ante fallas, los cuales se pueden identificar y aislar, deduciendo cual estado es el que no se dio y por lo tanto el sistema colocar al sistema en falla.

Tabla 3 Nodos de la red compleja.

Etiqueta	Descripción
E0	Reposo
E1	Motor banda uno, emitter, At scale entry
E2	Motor banda dos, At scale
E3	Sap
E4	At left entry, Motor selector izquierdo
E5	At right entry, Motor selector derecho
E6	At forward entry, Motor selector frontal
E7	At exit left, Motor banda 3
E8	At exit right, Motor banda 4
E9	At exit front, Motor banda 5

Metodología de detección

La detección de fallas consiste en conocer previamente la evolución del sistema a través del tiempo, con base en los eventos que se dan y que el controlador sabe que existen, tales como, la activación de un sensor o un botón. Es importante distinguir que, al activar un actuador o salida (motor, lámpara piloto), no es un evento que sea detectable por el controlador ya que el antes mencionado es una salida del controlador, pero realmente nunca se entera si el actuador o la salida se activó correctamente o no. Un ejemplo de esto es como enviar una carta a algún ser querido y no enterarse si este la recibió, solamente el remitente sabe que fue enviada, pero no sabe si es recibida. Mediante la información de tiempo de ciclos se conoce el comportamiento del sistema de manufactura, es decir, se sabe cada cuanto tiempo los eventos se sucederán. Para el caso de estudio se proponen los tiempos de ciclo de la tabla 4, en la cual se aprecia el tiempo en que se desata un evento en el sistema.

Tabla 4 Lista de tiempos y eventos.

Evento	Tiempo
BA	60 segundos al arrancar el sistema
At scale entry	5 segundos después de BA
At scale	3 segundos después de At scale entry
Sap	5 segundos después de At scale
At left entry	5 segundos después de Sap
At forward entry	5 segundos después de Sap
At right entry	5 segundos después de Sap
At exit left	7 segundos después de At left entry
At exit front	7 segundos después de At forward entry
At exit right	7 segundos después de At right entry

Todos los tiempos descritos por eventos estarán sucediendo siempre y cuando el predecesor suceda.

Con base en el modelo de red compleja anterior se obtiene la siguiente matriz de adyacencia de la ecuación 1 [Barabási, 2015].

$$A_{ij} = \begin{matrix} & \begin{matrix} E0 & E1 & E2 & E3 & E4 & E5 & E6 & E7 & E8 & E9 \end{matrix} \\ \begin{matrix} E0 \\ E1 \\ E2 \\ E3 \\ E4 \\ E5 \\ E6 \\ E7 \\ E8 \\ E9 \end{matrix} & \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \end{matrix} \quad (1)$$

En la matriz anterior se observa la relación que tienen los nodos de la red, elementos dados por la posición (i, j). Por columnas se trata de los nodos que están conectados con otros, siendo la columna el origen del link, por ejemplo, en la columna del E0 en todos los elementos de la matriz aparecen 0, ya que ese link no existe, excepto en el segundo elemento (2,1), este link está dado por la aparición de el botón de arranque, el cual es el intermediario entre la etapa cero (E0) y la etapa uno (E1).

Es importante saber que el uso de la matriz de adyacencia tiene un comportamiento dinámico, la cual cambia a través del tiempo, en caso de que el botón de arranque esté en falla, se apreciará un cambio de valor en el elemento (2,1), esto será identificado por el algoritmo y se pondrá el sistema en estado de falla. En la experimentación, se utiliza como caso de estudio el sistema de clasificación por peso, figura 1. Para realizar el control y el algoritmo de búsqueda se utilizó LabVIEW [National Instruments, 2017], el cual tiene el siguiente panel frontal, véase figura 9. En figura 9 se muestra ubicación del botón de arranque que inicializa el sistema de manufactura 4.0 propuesto, en la parte inferior, un indicador de estados y a un costado se encuentra la matriz de adyacencia, en la cual se logrará apreciar la evolución de esta a través del tiempo. Finalmente, en la parte inferior derecha, se encuentra el botón de paro.



Figura 9 Panel de control del proceso.

El panel posterior tiene dos funciones principales, una de ellas es realizar el control (como en cualquier empresa que ya tendría su sistema funcionando) y la otra es realizar la búsqueda e identificación de los elementos que son eventos detectables ante fallas descritas anteriormente, figura 10. Todo el programa está concentrado en una estructura tipo While, esta se encarga de ciclar infinitamente el sistema hasta que un botón de paro o un error sea detectado.

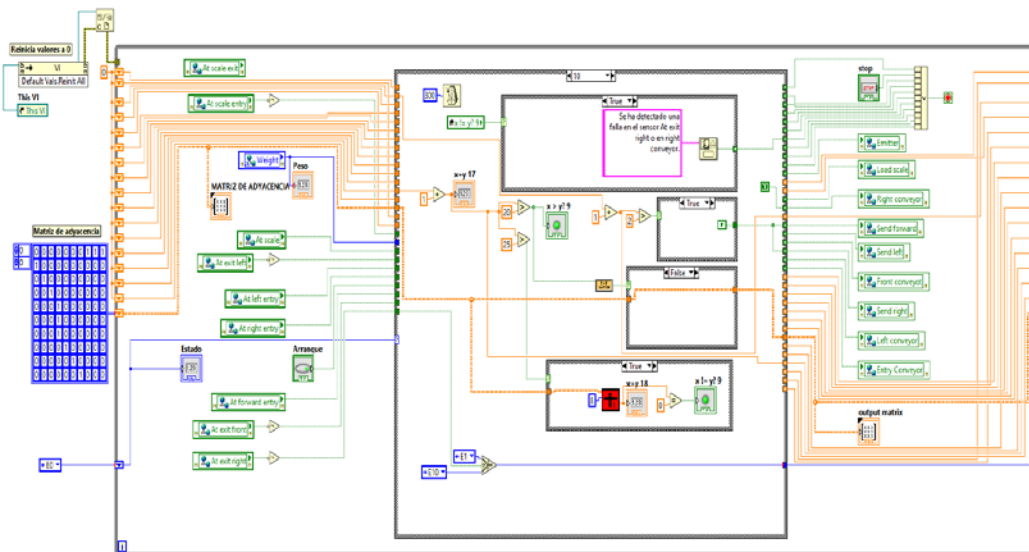


Figura 10 Panel de posterior.

Programa de control general

La aplicación de esta investigación consiste en simplemente evaluar un sistema ya funcional, por tal motivo para realizar las pruebas del algoritmo de detección de

fallas y la trasmisión de datos de forma remota, se necesita un sistema que funcione normalmente. Es importante mencionar que en esta etapa solamente se hablará de los elementos de control, aún no se indagará en el algoritmo de detección de fallas. En la figura 11, en la parte superior es una función la cual reinicia todos los valores a 0, esto hace referencia a que el sistema se reinicia totalmente, y que todos los sensores inmiscuidos olvidan el estado en el que estaban y comienzan a empezar un nuevo proceso. Por otro lado, la parte que dice Matriz de adyacencia que contiene elementos enteros de 0 y 1 y está coloreado de azul, es la matriz que se consigue al modelar estructuralmente el sistema de manufactura, misma que se presenta en la figura 8 de este documento.



Figura 11 Parte uno del programa de control.

La parte final que es un selector de estados (elemento azul con la leyenda E0) que se coloca para decirle al sistema cual es el estado de reposo, o el estado inicial en el que el sistema comenzará cuando este sea iniciado.

La figura 12 concentra los elementos de entrada del sistema empleado, tales como, los sensores ópticos, sensor de peso, botón de arranque y otros. Algunos de ellos contienen una compuerta tipo Not (marcados con un rectángulo color azul), esto debido a que la naturaleza del sensor es del tipo NC (normalmente cerrado) al agregar la compuerta Not, niega la naturaleza del sensor haciéndolo ahora NA (normalmente abierto). Los elementos con un rectángulo verde son indicadores, al estar conectados a alguna línea o cable estos sólo miden el valor que pasa por ese cable, por ejemplo, el indicador con la leyenda peso está midiendo el valor recabado por el sensor de peso.

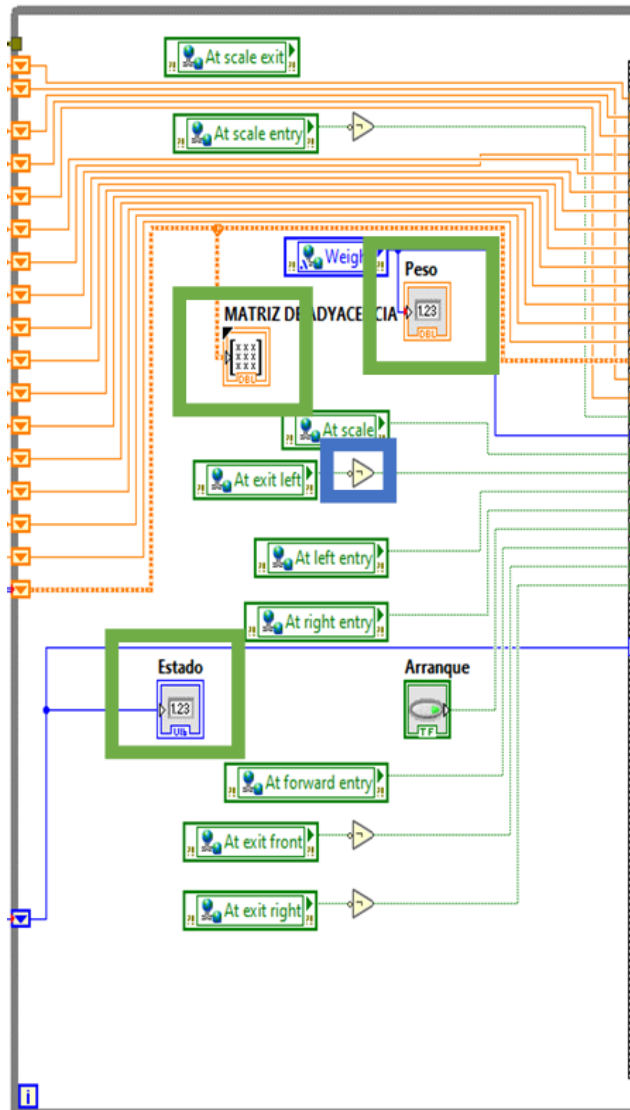


Figura 12 Parte dos del programa de control.

El indicador de matriz de adyacencia la muestra en el panel frontal para poder apreciar y verificar la evolución de esta, y finalmente el indicador de estado, registra el estado en el que se encuentra el sistema, estos estados son mayormente detallados en tabla 3. La figura 13 consta de todos los estados posibles del sistema, pero en cuestión de control, el rectángulo café que está en la parte superior de la figura tiene un número 10, el cual indica que el sistema se encuentra en estado 10.

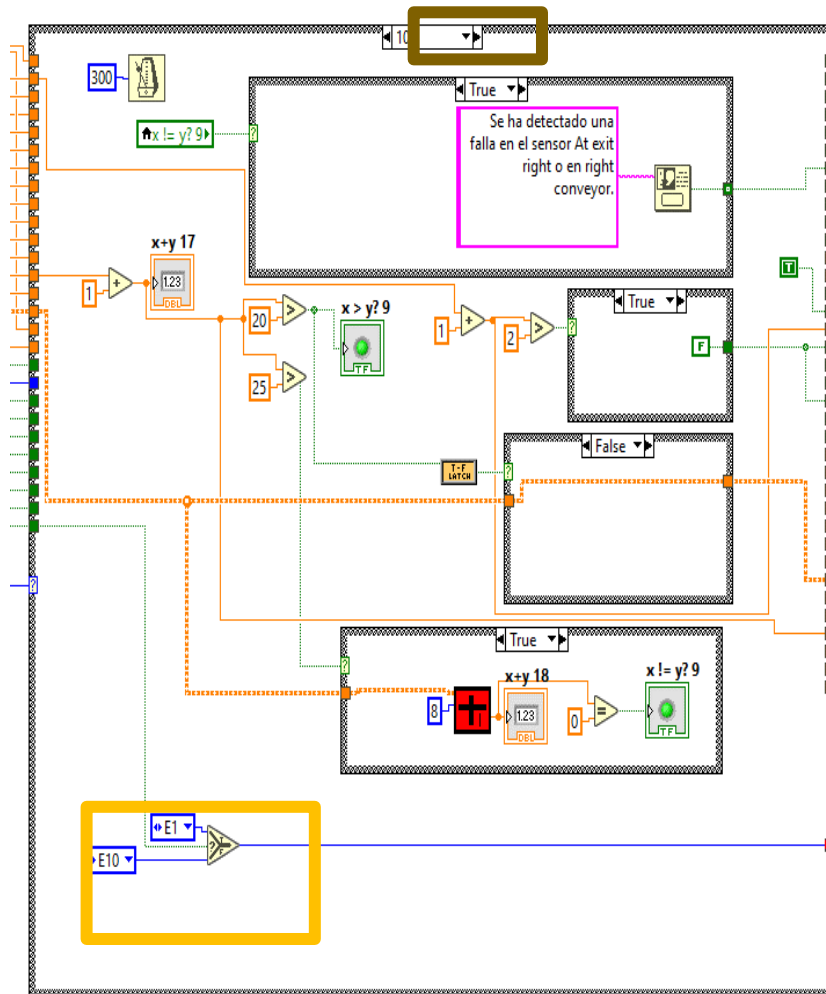


Figura 13 Parte central del programa de control.

En la parte inferior el rectángulo amarillo, integra un selector de estados, al que está conectado el elemento de transición como por ejemplo podemos decir que para ir del estado 0 al estado 1 se necesita que se active el botón de arranque, por lo tanto este selector revisa si ya fue activa, cuando este sucede, va al estado uno, y cuando

no sucede el botón de arranque, se queda en el mismo estado hasta que suceda algo contrario. El resto de las estructuras usadas no son explicadas en este espacio debido a que pertenecen al algoritmo de búsqueda. Algunos de los casos usados en el programa de control se muestran en la figura 14. El resultado final del programa de control se adjunta en el siguiente link, allí se encuentra el programa completo para el software LabVIEW versión 2017:

https://drive.google.com/drive/folders/1ITiLol-OpCGqGP1cbQ-wVFux3_SEebOB?usp=sharing

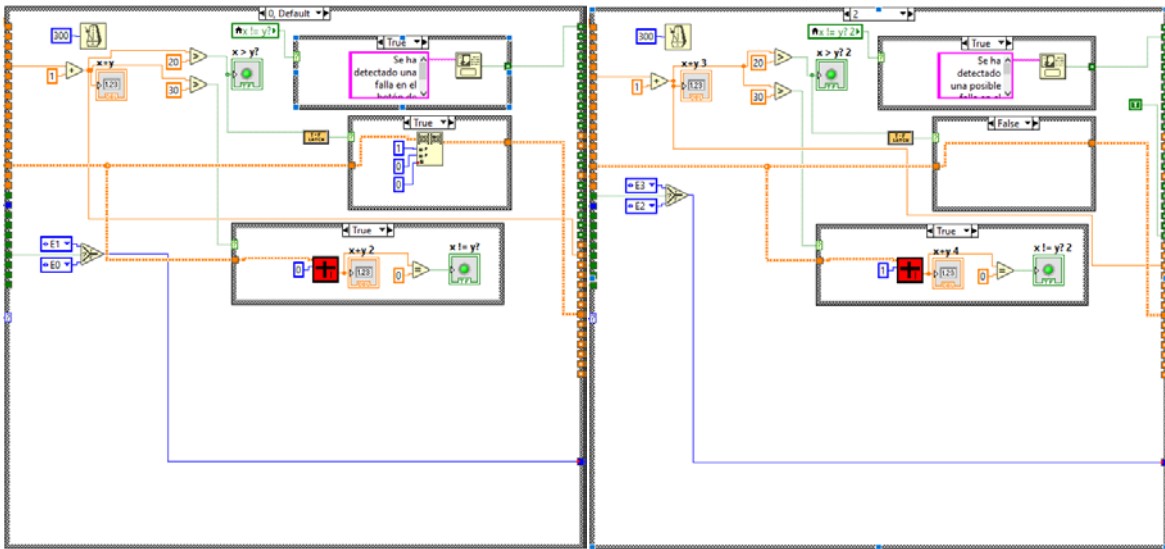


Figura 14 Otros casos utilizados.

Finalmente, la figura 15 en el rectángulo azul grisáceo integra las salidas del sistema o los elementos de acción, como lo son las bandas de los motores, así mismo los que generan y eliminan cajas. El programa de control es indispensable para poder hacer las pruebas debido a que la investigación parte de un sistema totalmente funcional, el cual es propenso a fallas por los elementos que están interconectados entre sí. Se pretende que funciona correctamente en todo instante del tiempo, pero dado que todos los elementos inmiscuidos son variantes en el tiempo, esto quiere decir que sufren desgaste, por tal motivo están expuestos a fallar. Con base en esta consideración todo sistema de cualquier naturaleza puede ser monitoreado ante fallas. La parte superior de la imagen aún no es explicada debido a que pertenece al algoritmo de detección de fallas.

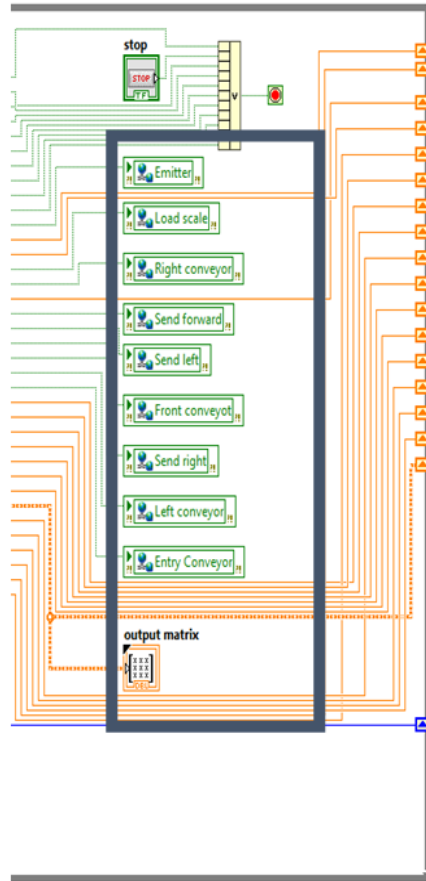


Figura 15 Parte final del programa de control.

Programa con algoritmo de detección de fallas

El algoritmo de detección está incluido en el programa anterior de LabVIEW y funciona a la par del programa de control, no quiere decir que así sucederá en una implementación real, aquí es necesario ya que se necesita tener un sistema funcionando el cual será evaluado. Para un sistema real se parte de que el sistema ya contiene sus elementos de control, como un PLC que ya hace que funcione el sistema de manufactura, lo que se haría es monitorear los links y mediante esta actividad se pueden diagnosticar fallas, la aplicación se describe en una sección más adelante.

Algoritmo de detección:

- Inicio
- Introducir la matriz de adyacencia según el modelado en red compleja.
- Guardar el estado en el que se encuentra el proceso (X_n).

- Identificar el estado siguiente a ese estado y el tiempo en que tarda en darse el evento (X_{n+1}).
- Iniciar un temporizador de valor ($t_n + 1$), el cual será el tiempo que normalmente se tarda el evento en darse más uno.
- Al finalizar el temporizador buscar en la posición de la matriz de adyacencia correspondiente al proceso el valor esperado.
- Si es valor es distinto al esperado, identificarlo como falla del sistema y detener el sistema. Si el valor es el esperado, se continúa el proceso.
- Fin.

Retomando laparte uno del programa, en figura 16 se observa matriz de adyacencia, debe ser introducida con base en modelado estructural de la red compleja.

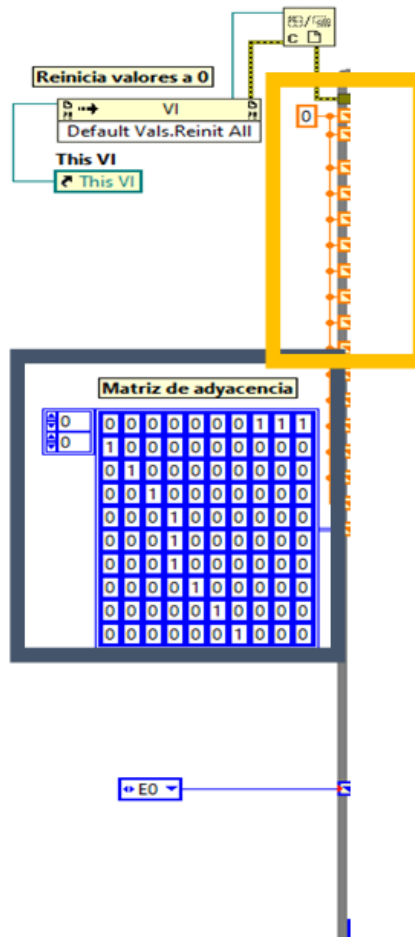


Figura 16 Parte uno del programa de detección de fallas.

Con un rectángulo de color amarillo se marca el número cero y varios cables que van conectados a este elemento, los cables son útiles para iniciar los temporizadores en cero. La parte dos del programa sólo contiene los cables que van desde la parte uno hasta la parte central del programa, los cuales transportan los valores de 0 para los temporizadores, figura 17.

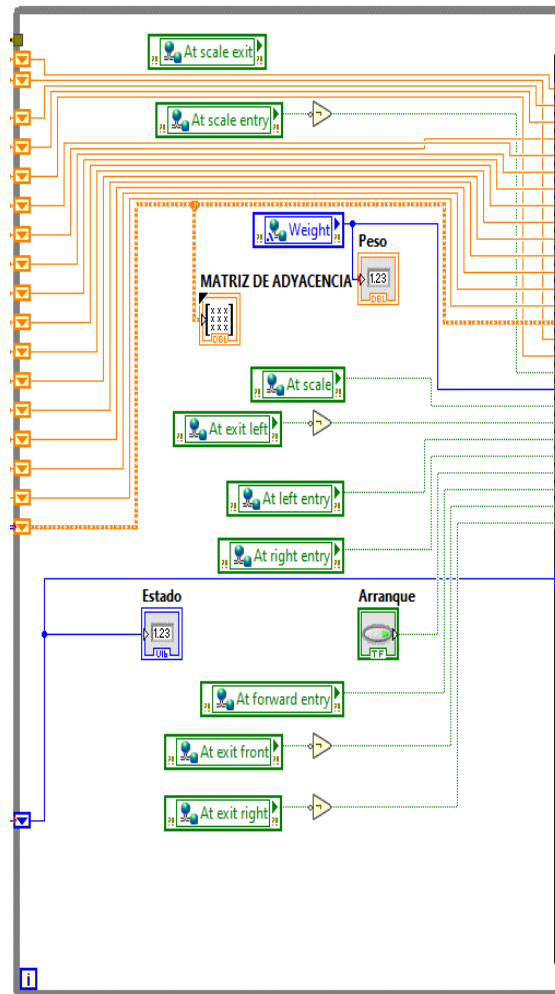


Figura 17 Parte dos del programa de detección.

En la parte central del programa cada caso de la estructura case utilizada tiene cuadro estructuras tipo case, que tienen como opciones verdadero y falso (True o False). El rectángulo color amarillo presenta un cuadro de dialogo para el usuario cuando se ha detectado una falla. El bloque color rojo es una función que agrega un cero cuando ya se excedió el tiempo y el link que debió haber sucedido en tiempo

y forma, no se dio (por ejemplo, cuando pasaron 60 segundos propuestos y aún no se activa el botón de arranque, esta parte del programa agrega un cero a la matriz de adyacencia en donde corresponde según el modelado estructural. Por último, la función final busca los ceros en la matriz de adyacencia la cual sólo busca donde corresponde el estado, si el valor es 1, no hará nada, pero si es igual a 0, activara el cuadro de dialogo para informarle al usuario de la falla encontrada, figura 18.

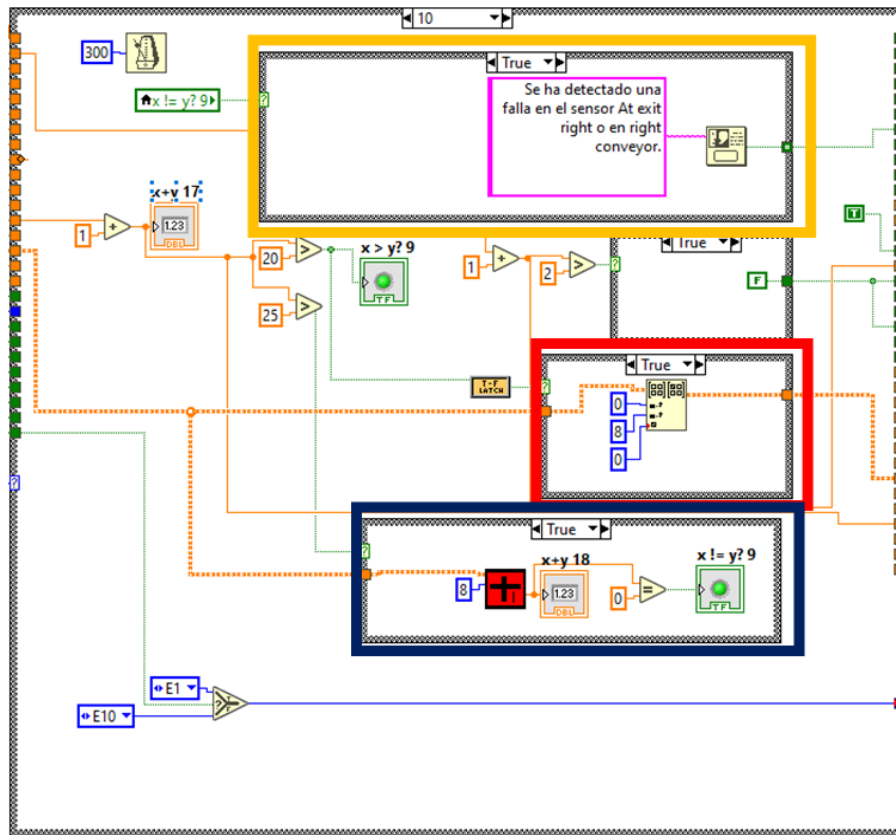


Figura 18 Parte central del programa de detección.

El programa se detiene automáticamente cuando los tiempos son excedidos y los eventos esperados no se dan, en el cuadro negro se observa un elemento con la leyenda stop, es el botón de paro general del programa, este a su vez está conectado con los cables verdes que se originan por el cuadro de dialogo que informa las fallas, todos estos están conectados a una compuerta tipo OR que se encarga de detener por completo el programa si se activa cualquier elemento conectado a la compuerta OR, figura 19.

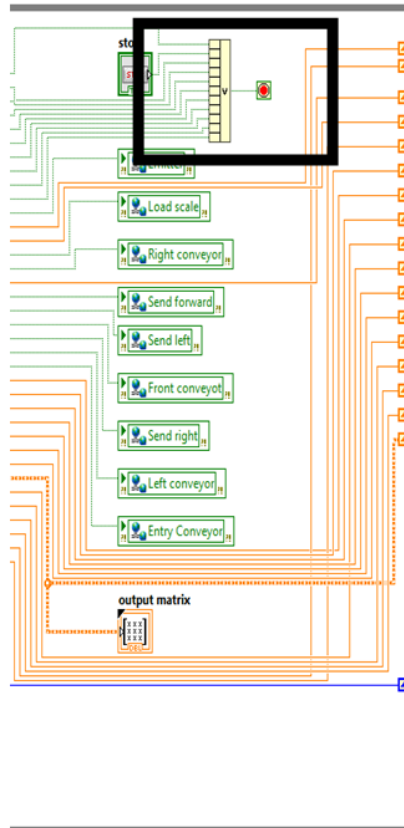


Figura 19 Parte final del programa de detección.

Conexión Factory IO y LabVIEW

La interconexión de los softwares LabVIEW y Factory IO se hace mediante un protocolo de comunicación tipo Modbus TCP/IP Server, dicha conexión y configuración está documentada en el siguiente video (fuente propia): <https://www.youtube.com/watch?v=H8aHu48QcFA&list=PLRDEvn0tBzEQtQx-T61OiIB1E7ESyRDio>.

Simulación de elemento en falla

Con ayuda del software Factory IO se realiza el proceso normal, en el que una caja de tamaño grande es clasificada, esta viaja por el centro de las bandas transportadoras, figura 20. Es importante mencionar que se puede realizar la detección de fallas en cualquiera de los elementos de la red compleja planteada, pero para esta demostración desconectamos el sensor llamado At exit front que corresponde al enlace con el mismo nombre.

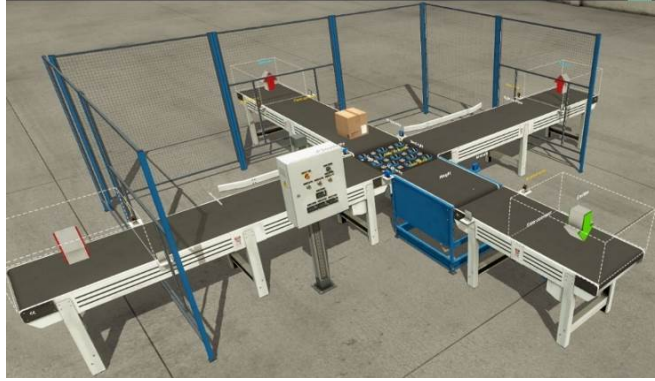


Figura 20 Clasificación caja grande.

Para poder generar la falla, ésta se hace en la parte final del proceso, en el cual la caja es retirada de manera forzada (figura 21), referencia marcada con una flecha negra.

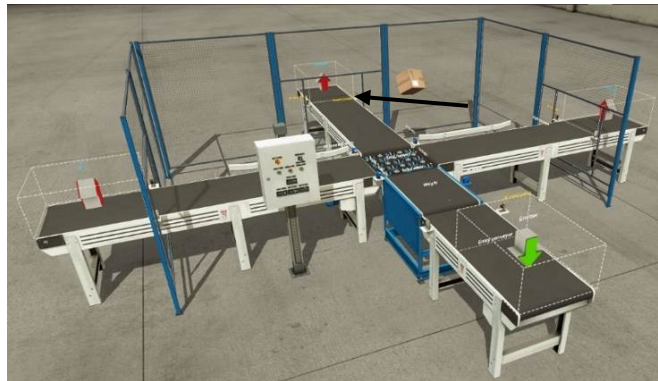


Figura 21 Caja retirada.

Dado el último evento que registró el sistema fue la lectura del sensor At forward entry, el sistema espera ver en la matriz de adyacencia en la posición (1, 10), un número uno que es la existencia del link dado por el siguiente evento, la lectura del sensor At exit front. Al no suceder esto último, el algoritmo puede concebir al sistema en falla, la cual es diagnosticada como falla del sensor At scale front o de la banda transportador perteneciente a esa línea (motor banda 5), figura 22.

En la figura 23, se puede observar la usencia del valor en uno, cuando el evento no es detectado a tiempo por el sistema. Por lo tanto, se detiene el programa y se le avisa al usuario las posibles causantes de la falla. El proceso de detección de fallas

es de suma importancia debido a que no agrega sensores al proceso ya establecido, ya que basado en el historial de los eventos (matriz de adyacencia) se puede hacer el modelado en red compleja de cualquier sistema de manufactura y con esto facilitar la búsqueda y diagnóstico de fallas.

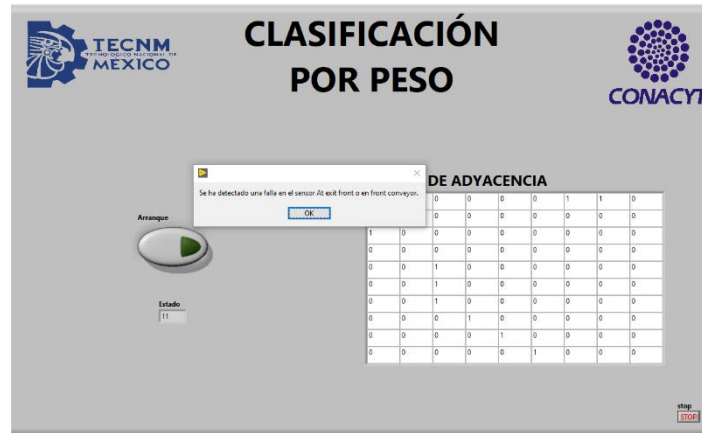


Figura 22 Diagnostico de falla.

0	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	1	0	0

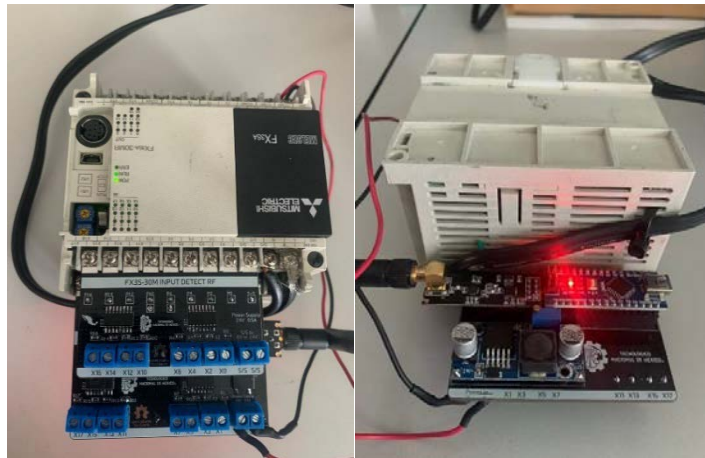
Figura 23 Matriz de adyacencia en falla.

Monitoreo y detección remota en tiempo real

En el resultado de la detección de fallas en tiempo real, se partió del programa de control que ya se tenía sólo se desconectaron los elementos de entrada en la parte principal del programa y se sustituyeron por los valores recabados por el Arduino que se transmiten vía serial. Esto se hizo porque las entradas ahora están de forma física para este apartado de la simulación.

Para las pruebas en tiempo real se utilizó un PLC Mitsubishi FX3SA-30MR, el sistema de monitoreo remoto (transmisor) consiste en una placa de circuito impreso

colocada en las terminales del PLC junto con un Arduino mini y un módulo de radio NRF24L01 con conector para Antena, como se muestra en la figura 24 inciso a y b. Se decidió utilizar el módulo de radio NRF24L01 ya que tiene un alcance de 1000 metros en campo abierto, con esto es más que suficiente para una PyME (Pequeñas y medianas empresas) del país [secretaría de economía, 2020]. En la estación receptora (figura 25), se utiliza un Arduino Pro mini y un módulo NRF2401 sin antena debido a que no es necesaria, ya que el largo alcance está dado por el trasmisor.



a) Superior

b) Inferior

Figura 24 Placa de control y monitoreo remoto trasmisor y receptor.

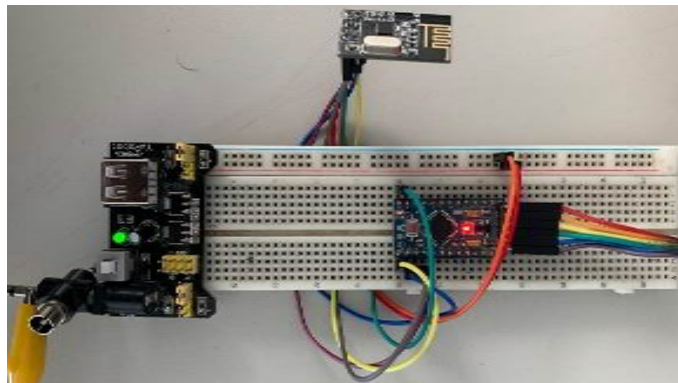


Figura 25 Estación receptora.

El programa de la estación de transmisión identifica cuando se activa una entrada del PLC, posteriormente, envía dicha información al sistema receptor, este último su función consiste en obtener toda la información enviada por el trasmisor y mostrarlos

a la computadora a través de una comunicación serial. Cuando la placa de transmisión detecta que se ha activado una entrada, automáticamente manda la señal de cual de todas las entradas fue activada mediante el número 1, cuando está aún no está activada presenta un 0.

En el programa de control se modificaron dos partes importantes una de ellas es el recabado de datos vía serial. En el rectángulo azul se aprecia la inicialización serial de la estación receptora conectada al COM 12, figura 26.

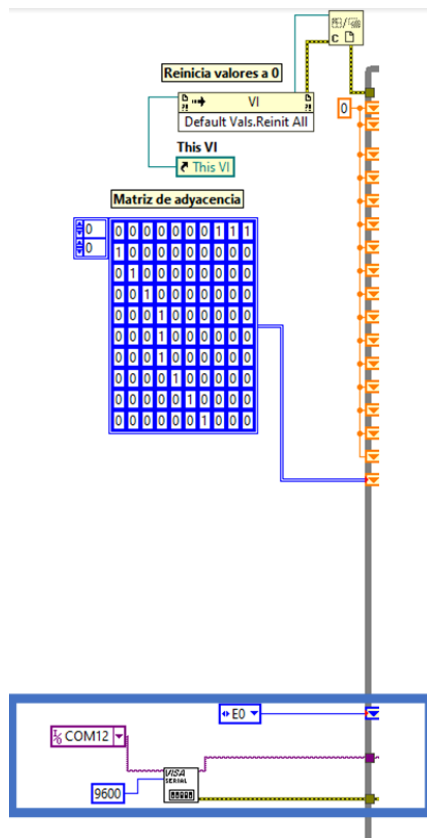


Figura 26 Parte uno del programa, conexión serial.

En la parte dos del programa se agregaron las funciones para leer los datos obtenidos vía serial, figura 27 rectángulo color azul. Los datos leídos por la función de LabVIEW son distribuidos a un VI de color rojo que se encuentra en el rectángulo de color negro, dicha función es utilizada para separar el paquete de valores que es mandado por la estación de trasmisión. Posteriormente se comparan con el valor uno y se conectan a las entradas como normalmente se tendrían. Cabe destacar

que el resto del programa es idéntico ya que la prueba de detección de fallas en tiempo real sólo es omitir las entradas proporcionadas por el software Factory IO, estas son sustituidas por las entradas físicas que se utilizarán en el siguiente apartado.

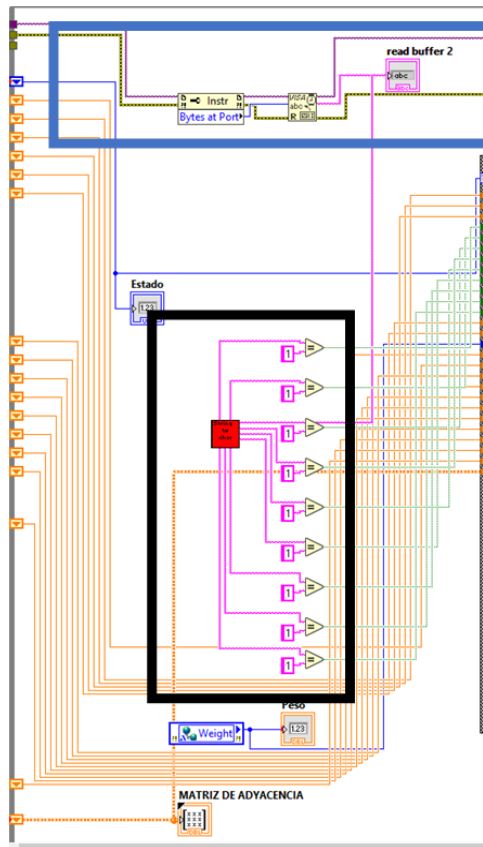
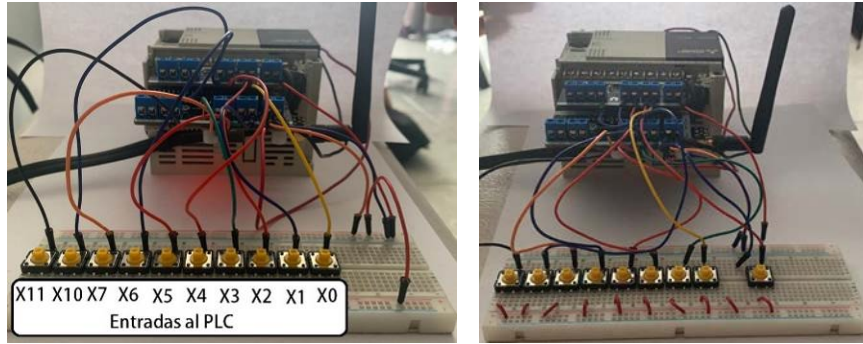


Figura 27 Parte dos del programa comparación de datos.

Detección de fallas en tiempo real

Para la detección de fallas en tiempo real se emula mediante botones externos las entradas del sistema propuesto (tales como botón de arranque, sensores de límite). Para generar una posible falla se desconecta de la entrada X1 que corresponde al sensor At Scale Entry, figura 28b. En la tabla 5, se observa la equivalencia de las entradas del sistema propuesto. El salto de la entrada X7 a la X10 es porque así está constituido el PLC (no tiene entrada X8 y X9). La falla será causada por la ausencia del link llamado At Scale Entry, como se muestra en la figura 29.



a) Botones como entradas

b) Falla en X1

Figura 28 Entradas simuladas.

Tabla 5 Tabla de entradas al PLC.

Etiqueta	Entrada PLC
BA	X0
At scale entry	X1
At Scale	X2
Sap	X3
At left entry	X4
At forward entry	X5
At right entry	X6
At exit left	X7
At exit front	X10
At exit right	X11

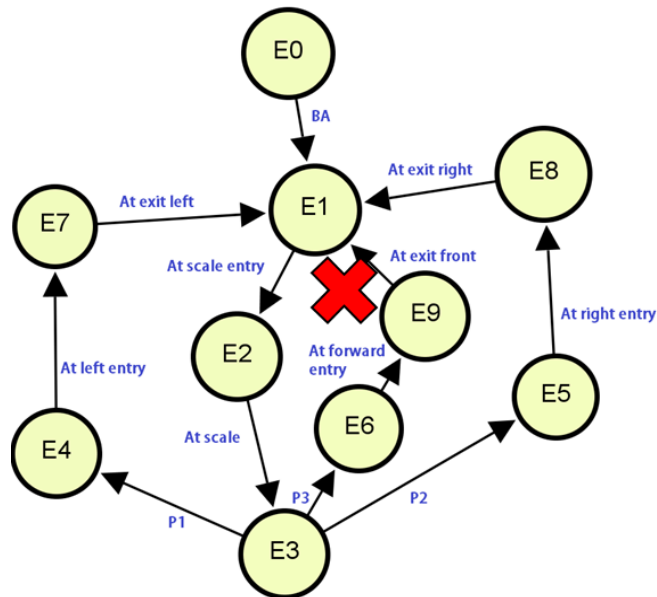


Figura 29 Red compleja en falla.

La secuencia de datos que se reciben en esta prueba son documentados por el monitor serial de Arduino de la estación de recepción los cuales se indican en la figura 30. Se observa que se da el botón de arranque y el siguiente evento esperado es que el sensor At Scale Entry no sucede por lo tanto después de que se cumple el Timmer, el algoritmo puede concebir al sistema en falla.

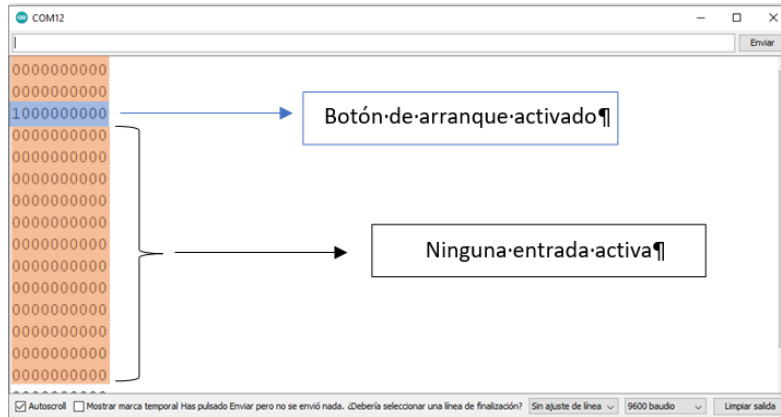


Figura 30 Datos enviados vía serial.

En el momento dado que el sistema espera la señal del sensor At Scale Entry para proceder al siguiente estado del sistema, este identifica en la matriz de adyacencia el elemento en 0 (Figura 31), mismo que reporta al usuario la falla identificada y los posibles causantes de la misma, las opciones de la falla es que la banda transportadora (B1) no funciona, que el Emitter no funciona o bien que el sensor At Scale Entry no mando la señal. La Solución a este problema es que el sensor At Scale Entry se encuentra desconectado.



Figura 31 Falla en tiempo real.

3. Resultados

Los resultados obtenidos hasta este momento en la investigación han sido:

- Modelo de red compleja desde el enfoque de un sistema de manufactura de la industria 4.0.
- Detección de fallas en un sistema de manufactura 4.0 en el cual no se agregan sensores extras, esto es de suma importancia ya que no agrega gastos en sensores o monitoreo.
- Metodología de detección de fallas versátil para sistemas de manufactura capaz de trabajar con pocos elementos, como es mostrado aquí, o miles de ellos.
- Algoritmo de detección de fallas de un sistema complejo dinámico y de evolución en tiempo real.
- Simulación de sistema de manufactura funcional en software Factory IO.
- Conexión entre software LabVIEW y Factory IO para el control, monitoreo y detección y localización de fallas del sistema de manufactura.
- A diferencia de [Reyes, 2015] y [García, 2015] que agregan por cada rama de la red compleja un sensor para monitorear el estado completo, en este artículo se agregan cero sensores.

4. Discusión

Gracias a la metodología de detección de fallas planteada aquí, se logra establecer una técnica viable y barata de implementar por modelado en redes complejas, ya que lo mostrado en [Reyes, 2015] y [García, 2015], utiliza sensores en diferentes puntos de la red, si se toma esta inventiva para algún sistema de manufactura de miles elementos, el resultado sería tener miles de sensores extras también. Por lo tanto, no es viable.

La idea de realizar la detección de falla en los links es tomada de [Dhal, 2015] y [Preciado, 2015] aunque la técnica empleada es diferente, ya que este artículo se basa en los tiempos de evolución del sistema y los antes mencionados en algunos puntos de observación y estimaciones de comportamiento de la red compleja.

5. Conclusiones

Los objetivos principales han sido cumplidos con éxito, tales como realizar el modelado en red compleja de un sistema de manufactura de la industria 4.0. Del mismo modo realizar una conexión entre el software Factory IO y LabVIEW para realizar simulaciones y pruebas. Y como parte importante se obtiene un algoritmo capaz de realizar la detección y localización de la falla, parte medular de la investigación es que se logra realizar la detección de una falla basado en una metodología de ciclos perteneciente al área de la ingeniería industrial, con este avance sólo se migra la información del modelo en red compleja y sin sensores extras al proceso, se identifican las fallas, sin importar la cantidad de elementos que tenga el sistema de manufactura real.

Actualmente aún se buscan otros resultados tales como el monitoreo de forma remota para poder identificar fallas a distancia y tomar una decisión acerca de la falla (control), esto se implementará con tecnología IoT (Internet of things) basada en las tecnologías utilizadas por [Heynicke,2018].

6. Bibliografía y Referencias

- [1] Barabási, A. L. Network science. Graph Theory, 3-29, chapter 2. Cambridge University Press, 2015. <http://networksciencebook.com/>.
- [2] Bonacina, F., Corsini, A., Cardillo, L., & Lucchetta, F., Complex Network Analysis of Photovoltaic Plant Operations and Failure Modes. Rome, Italy. Mayo, 2019.
- [3] Dhal, R., Torres, J. A., & Roy, S. Detecting link failures in complex network processes using remote monitoring. USA. Mayo, 2015.
- [4] García, J., Escobar, R. F., & Astorga, C. M. Diagnóstico de Fallas Múltiples en los Sensores de un Intercambiador de Calor Empleando Observadores Por modos Deslizantes Basado en el Algoritmo Super- Twisting. Cuernavaca, Morelos, Mayo, 2015.
- [5] Heynicke, R., Krush, D., Cammin, C., & Scholl, G. IO-Link Wireless enhanced factory automation communication for Industry 4.0 applications. Hamburg, Germany, Marzo, 2018.

- [6] Kleinberg, J., Sandler, M., & Slivkins, A. Network Failure Detection and Graph Connectivity. New York, USA. Julio, 2017.
- [7] Mastos, T. D., Nizamis, A., & Vafeiadis, T. Industry 4.0 sustainable supply chains: An application of an IoT enabled scrap metal management solution. Thessaloniki, Greece. Mayo, 2020.
- [8] National Instruments. (2017). LabVIEW (Versión 17.0): <https://www.ni.com/es-mx/support/downloads/software-products/download.labview.html#346254>.
- [9] Preciado, V. M., & Rahimian, M. A. Failure Detection and Isolation in Integrator Networks. Chicago, USA. Julio, 2015.
- [10] Real Games. (2020). Factory IO (Versión 2.2.3- Siemens Edition): <https://factoryio.com/>.
- [11] Reyes, J., Ruiz, E., Orozco, J. L., & Castañeda, L. A. Detección de Fallas en un Sistema de Manufactura Flexible desde el enfoque de la teoría de las Redes complejas. Aguascalientes, México. Diciembre, 2015.
- [12] Shang, Y. Localized recovery of complex networks against failure. Shanghai, China. Julio, 2016.
- [13] Tjahjono, B., Espluegues C., Ares E., & Pelaez G. What does Industry 4.0 mean to supply chain? Procedia Manufacturing. Vigo, España. Junio, 2017.
- [14] Zhao, C., Gao, F., Yao, Y., & Cao, Z. Fault Detection, Isolation, and Prognosis for complex system. USA. Enero, 2018.