UNIVERSIDAD AUTÓNOMA DE CHIHUAHUA

FACULTAD DE INGENIERÍA

SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

DESARROLLO DE UN EMULADOR DE UN SISTEMA DE PRODUCCIÓN DISCRETA CON BASE EN IMRP UTILIZANDO LA OPERACIÓN DIRIGIDA POR MODELO

POR:

CARLOS IVÁN ONTIVEROS ROACHO

TESIS PRESENTADA COMO REQUISITO PARA OBTENER EL GRADO DE

MAESTRO EN INGENIERÍA EN COMPUTACIÓN

CHIHUAHUA, CHIH., MÉXICO **JUNIO DE 2017**

"Desarrollo de un emulador de un sistema de producción discreta con base en iMRP utilizando la operación dirigida por modelo".

Tesis presentada por Carlos Iván Ontiveros Roacho como requisito parcial para obtener el grado de Maestro en Ingeniería, ha sido aprobada y aceptada por:

M.I. Javier González Cantú
Director de la Facultad de Ingeniería

Dr. Fernando Rafael Astorga Bustillos
Secretario de Investigación y Posgrado

M.S.I. Karina Rocío Requena Yáñez
Coordinador(a) Académico

Dr. José Eduardo Acosta Cano de los Ríos
Director(a) de Tesis

JUNIO 2017

Comité:
Dr. José Eduardo Acosta Cano de los Ríos
Dr. Pedro Rafael Acosta Cano de los Ríos
MI. José Manuel Siqueiros Morales

Carlos Iván Ontiveros Roacho
C. Campo del Manzanar 9710,
Col. Campo Bello, C.P 31124,
Chihuahua, Chihuahua.

JUNIO 2017

UNIVERSIDAD AUTÓNOMA DE CHIHUAHUA

Facultad de Ingeniería

15 de junio de 2017

ING. CARLOS IVÁN ONTIVEROS ROACHO

Presente

En atención a su solicitud relativa al trabajo de tesis para obtener el grado de Maestro en Ingeniería en Computación, nos es grato transcribirle el tema aprobado por esta Dirección, propuesto y dirigido por el director **Dr. José Eduardo Acosta Cano de los Ríos** para que lo desarrolle como tesis, con el título: **"DESARROLLO DE UN EMULADOR DE UN SISTEMA DE PRODUCCIÓN DISCRETA CON BASE EN iMRP UTILIZANDO LA OPERACIÓN DIRIGIDA POR MODELO".**

ÍNDICE

Facultad de Ingeniería
Circuito No. 1, Campus Universitario 2
Chihuahua, Chih. C.P. 31125
Tel. (614) 442-9500 www.fing.uach.mx

UNIVERSIDAD INTERNACIONAL

UNIVERSIDAD AUTÓNOMA DE CHIHUAHUA

Facultad de Ingeniería

CAPÍTULO 4. DESARROLLO DE LA SOLUCIÓN PROPUESTA

4.1 SISTEMA INFORMÁTICO DE CONTROL

4.2 APLICACIÓN DEL CALCULO OEE

CAPÍTULO 5. VALIDACIÓN Y RESULTADOS DE LA SOLUCIÓN PROPUESTA

5.1 VALIDACIÓN DEL SISTEMA LANZADOR A NIVEL LABORATORIO

5.2 SISTEMA LANZADOR DE ÓRDENES EN UN PISO REAL DE PRODUCCIÓN

5.3 APLICACIÓN DE REPORTES Y CÁLCULO DE OEE EN UN PISO REAL DE PRODUCCIÓN

5.4 ANÁLISIS DE RESULTADOS

CAPÍTULO 6. CONCLUSIONES Y RECOMENDACIONES

Referencias

Currículum Vitae

Solicitamos a Usted tomar nota de que el título del trabajo se imprima en lugar visible de los ejemplares de las tesis.

A T E N T A M E N T E
"naturam subiecit aliis"

EL DIRECTOR

M.I. JAVIER GONZÁLEZ CANTÚ

FACULTAD DE INGENIERÍA U.A.CH. DIRECCIÓN

EL SECRETARIO DE INVESTIGACIÓN Y POSGRADO

DR. FERNANDO RAFAEL ASTORGA BUSTILLOS

Dedicatoria

A mis padres
Mis más grandes héroes, símbolos de admiración y respeto.

A mis hermanos Marleny y Luis C.
Que sepan que con esfuerzo, dedicación y constancia pueden lograr lo que se propongan.

Agradecimientos

En primer lugar a Dios, que me ha brindado una vida llena de alegrías y aprendizaje, situando los medios que fueron necesarios para que yo llegara hasta donde estoy.

A mis padres Carlos y Soco les agradezco infinitamente por su grandes muestras de amor, por ser mi ejemplo a seguir, por ese apoyo económico y especialmente moral, por no permitirme dudar en ningún momento y por enseñarme que puedo volar tan alto como lo desee.

A mis hermanos Luis y Marleny por preocuparse siempre por mí y por ser mi motivación de ser mejor persona cada día.

Al Consejo Nacional de la Ciencia y Tecnología (CONACYT) y a por el apoyo económico otorgado durante los dos años de estadía en la maestría.

A la Universidad Autónoma de Chihuahua, específicamente a la Facultad de Ingeniería por ser mi casa de estudios profesionales durante seis años.

A la secretaria de investigación y posgrado de la Facultad de Ingeniería de la Universidad Autónoma de Chihuahua por abrirme las puertas y darme la oportunidad de cursar este posgrado.

A mi director de tesis y también profesor de clases el Dr. José Eduardo Acosta Cano de los Ríos le agradezco enormemente por permitirme formar parte de su equipo de trabajo y dar seguimiento a su investigación doctoral, por aportar sus conocimientos y enseñanzas para formarme como profesional.

A mis Asesores y profesores el Dr. Pedro Acosta Cano de los Ríos y M.I. Manuel Siqueiros quienes con su gran experiencia me aportaron grandes enseñanzas de la profesión.

Al M.I. David Maloof Flores y ex coordinador de la maestría en ingeniería en computación por depositar su confianza en mí y darme la oportunidad de cursar este posgrado y por su incondicional apoyo durante los dos años de estadía en la maestría.

A Almita Carrera le estoy profundamente agradecido por las grandes muestras de cariño, por motivarme a superarme siempre, por su gran nobleza y por tener ese espíritu de ayudar a los que más lo necesitan.

A mis amigos Cubanos y compañeros de clase el M.I. Javier Gonzáles y el Ing. Evelio Clavel por compartir sus conocimientos y su cultura conmigo y por convertir cada desvelada en las mejores reuniones de la historia.

Al Ing. Miguel A. González y el Ing. Marco A. González que como compañeros de tesis me apoyaron a lo largo de todo el proyecto aportando sus grandes ideas en el análisis de conceptos y en el desarrollo del proyecto; en especial a Miguel por ser mi amigo y compañero inseparable, por su gran paciencia demostrada y su gran sentido del humor.

A mis mejores amigos Horlando Robles, Rubén Domínguez, Raúl Cano, Marco Carreón y Milton Holguín por acompañarme fielmente en esta y todas las aventuras de mi vida.

A mis maestros de la maestría por sus enseñanzas, dedicación y entrega.

A todos los que han recorrido este camino conmigo, muchas gracias.

Resumen

En la actualidad las empresas de manufactura discreta buscan mejorar la manera en que se modelan los procesos así como también buscan soluciones para integrar nuevos equipos y medir el rendimiento de los mismos en un piso de producción con el menor esfuerzo posible, para ello implementan sistemas de control capaces de controlar e integrar nuevos equipos de producción a las células de manufactura, sin embargo el desarrollo de dichos sistemas de control no se basa en algún estándar sino que son llevados a cabo según las problemáticas que se presentan en el piso de producción.

En el presente proyecto de tesis se presenta el desarrollo de un sistema informático de control lanzador de órdenes de operación basado en la arquitectura de referencia ArquiTAM en la cual se definen claramente los módulos que lo componen y la interacción entre ellos utilizando los conceptos de acoplamiento débil. Dicho sistema se rige bajo la técnica de operación dirigida por modelo, es decir, es capaz de interpretar un modelo grafico denominado iMRP y con base en el modelo envía órdenes de operación a los equipos de producción. Además, para medir el rendimiento de los equipos se presenta el desarrollo de una aplicación capaz de generar reportes y calcular el métrico de desempeño OEE.

La solución propuesta es validada a nivel laboratorio simulando una estación de producción discreta integrando varios equipos al sistema de control. Además, se realizó un ejercicio de validación del sistema de control en una empresa de productos químicos del giro de producción por lotes donde se integraron varios equipos y se enviaron órdenes de operación con base en un modelo iMRP creado a partir de una receta de un producto químico. Finalmente se crearon reportes y se calculó el métrico OEE a varios equipos para medir el rendimiento de los mismos.

Palabras Clave: iMRP, ArquiTAM, Acoplamiento débil, Operación dirigida por modelo, OEE.

Abstract

At present, discrete manufacturing companies seek to improve the way processes are modeled as well as solutions to integrate new equipment and measure their performance on a production floor with the least possible effort. For this, they implement control systems able to control and integrate new production equipment to the manufacturing cells, however, the development of such control systems is not based on any standard rather they are carried out according to the problems that arise in the production floor.

In this thesis project is presented the development of a computer system of control launcher of operation orders based on the ArquiTAM reference architecture in which there are clearly defined the modules that compose it and the interaction between them using the concepts of weak coupling . This system is governed by the technique of model driven operation, that is, it is able to interpret a graphic model called iMRP and based on the model it sends operating orders to the production equipment. Moreover, to measure the performance of the equipment there is presented the development of an application capable of generating reports and calculate the OEE performance metric.

The proposed solution is validated at the laboratory level simulating a discrete production station integrating several equipment into the control system. In addition, a validation exercise of the control system was carried out in a chemical company of the batch production spin where several equipment were integrated into the control system and orders of operation were sent based on an iMRP model created from a chemical recipe. Finally, reports were created and the OEE metric was calculated to measure the performance of several equipment.

Keywords: iMRP, ArquiTAM, weak coupling, model driven operation, OEE.

Tabla de Contenido

Índice de Tablas

Índice de Figuras

Índice de Listados

CAPÍTULO 1. INTRODUCCIÓN

En las últimas décadas hemos sido testigos de un acelerado avance tecnológico en las empresas manufactureras, el gran aumento de la competencia y los cambios tan turbulentos que ocurren en el mercado han obligado a las empresas a optimizar la manera en que modelan sus procesos de producción y a desarrollar aplicaciones de control más flexibles para responder a las demandas de los clientes en el menor tiempo posible y a la vez aumentar exponencialmente la producción y la calidad de sus productos, sin embargo para satisfacer dichas necesidades las empresas modifican sus sistemas según las dificultades que van ocurriendo en el piso.

Para modelar el comportamiento de sistemas de fabricación discreta se crearon herramientas gráficas y matemáticas, destaca la técnica de redes de Petri desarrollada a finales de los años sesenta, que ayudan a modelar sistemas concurrentes, asíncronos y distribuidos (Murata, 1989). Sin embargo, existen sistemas de eventos discretos, cuyo modelado con redes Petri se vuelve complicado, por tal motivo, a la versión original se le han incorporado características que permiten diseñar sistemas más complejos (Llorens & Oliver, 2004).

Con base en redes de Petri han surgido otras herramientas para modelar sistemas de fabricación, como Grafcet, Sequential Function Charts (SFC), Procedure Function Chart (PFC), Integration Definition for Function Modeling (IDEF0), Unified Modeling Language (UML), etc.

Otro tipo de modelado de sistemas son aquellos basados en el modelo iMRP, con el cual, se ha trabajado desde inicios de la década pasada; dicho modelo permite la integración de distintos sistemas de producción de una manera práctica y fácil.

Por otra parte, las empresas de manufactura discreta además de buscar modelar de una manera más eficiente los procesos a nivel de piso de producción, también desean medir el rendimiento de los equipos, actualmente existen varios métricos para medir el desempeño de los equipos, sin embargo, uno de los más utilizados es el métrico OEE el cual es calculado utilizando aplicaciones informáticas.

El objetivo del presente proyecto es el desarrollo e implementación de un sistema informático lanzador de órdenes de producción que sea flexible respecto a los procesos, equipos y funciones a manejar en el sistema. Se emplea la técnica de operación dirigida por

modelo, tomando como base el esquema de modelado iMRP y el esquema de referencia ArquiTAM.

La presente tesis está organizada de la siguiente manera: En el capítulo dos, correspondiente al marco teórico, se explican los conceptos utilizados durante el desarrollo con el objetivo de adentrar al lector en algunos conceptos técnicos para apoyar la comprensión del contenido en general. Posteriormente, en el capítulo tres se presenta el planteamiento del problema donde se describe la problemática a abordada y la solución propuesta, así como los objetivos y la justificación de la realización del proyecto de tesis. El capítulo cuatro se enfoca en el desarrollo del sistema de control lanzador de órdenes y la aplicación para el cálculo del métrico OEE. En el capítulo cinco se presenta al lector la validación del sistema de control en un área de producción discreta a nivel laboratorio y también se muestra la aplicación del sistema lanzador y la aplicación del cálculo OEE en una empresa real en un ambiente de producción por lotes. Finalmente, se presentan las conclusiones del proyecto y el trabajo futuro a realizar, así como las referencias a artículos consultados para hacer posible la investigación.

CAPÍTULO 2. MARCO TEÓRICO

En el marco teórico que fundamenta esta investigación se encontrarán conceptos básicos de modelado de sistemas y se analizarán distintas maneras de modelar los procesos en un piso de producción. Además, se hace un análisis detallado de los niveles de la Arquitectura de referencia de talleres de manufactura (ArquiTAM) y se explican conceptos específicos como acoplamiento débil y operación dirigida por modelo. Finalmente se analiza detalladamente el parámetro de desempeño OEE el cual ayuda a medir el desempeño de los equipos en un piso de producción.

2.1 TÉCNICAS DE MODELADO.

Modelar de manera gráfica los procesos que se llevan a cabo en un piso de producción ayuda a simplificar la comprensión del comportamiento de las actividades que se realizan. En un modelo se abstrae la información de un piso de fabricación la cual incluye equipos, operaciones y la secuencia de las actividades en los procesos, etc.

A finales de los años sesenta se empezaron a crear herramientas gráficas y matemáticas que permiten modelar el comportamiento de sistemas de fabricación discreta, como son las redes de Petri, que permiten analizar, simular, controlar y evaluar el comportamiento de sistemas concurrentes, asíncronos y distribuidos (Murata, 1989), (Wilfried & Wolfgang , 2006).

Sin embargo, existen sistemas de eventos discretos cuyo modelado con las redes Petri tradicionales se vuelve complicado, por tal motivo, a la versión original se le han incorporado características que permiten diseñar y simular sistemas más complejos (Llorens & Oliver, 2004). Entre las extensiones que se han propuesto están las redes de Petri: temporizadas deterministas, temporizadas probabilísticas, estocásticas, coloreadas, coloreadas condicionales, difusas, generalizadas, con arcos inhibidores, con capacidad limitada, entre otras.

Con base en redes de Petri han surgido otras herramientas para modelar sistemas de fabricación, como Grafcet, Sequential Function Charts (SFC), Procedure Function Chart (PFC), Integration Definition for Function Modeling (IDEF0) y otros como Unified Modeling Language (UML).

Otro tipo de modelado de sistemas, son aquellos basados en el modelo iMRP, con el cual se ha estado trabajando desde inicios de la década pasada en la línea de investigación de

Automática e Informática Industrial del Instituto Tecnológico de Chihuahua y la Universidad Autónoma de Chihuahua. Se han realizado trabajos enfocados a la integración de distintos sistemas de producción como sistemas de fabricación discreta y de fabricación por lotes (clasificación hecha por la Sociedad Internacional de Automatización ISA), en los estándares ISA S88 e ISA95 en ambientes de manufactura.

2.1.1 Redes de Petri.

El concepto de redes Petri fue creado por el alemán Carl Adam Petri en su tesis doctoral "Comunicación con autómatas" que presentó a la facultad de Matemáticas de la Universidad de Bonn en julio de 1961 y defendió en junio de 1962 (Petri, 1962).

Una red de Petri es un tipo de grafo dirigido con un estado inicial denominado marcado inicial (M0), en el cual intervienen dos tipos de nodos, las "plazas" o "lugares" y las "transiciones", además, contiene arcos que tienen un peso específico y que se encargan de conectar a elementos de distinto tipo. Gráficamente, los lugares son representados mediante círculos, y las transiciones mediante rectángulos o barras. Una plaza puede contener un número entero, positivo o nulo, de tokens, un token se representa por un punto en el interior de la plaza. La evolución de estos tokens conforma el funcionamiento de la red, permitiendo representar sistemas dinámicos mediante redes de Petri, como se muestra en la figura 1 (Recuero & Alvarez, 1997).

Desde finales de la década de los sesenta la teoría de las redes de Petri comenzó a tener un desarrollo considerable en la industria manufacturera y se le ha dado usos muy variados en las ciencias de la computación, ya que son una herramienta de modelación gráfica y matemática que puede aplicarse en el modelado de varios tipos de sistemas (Wilfried & Wolfgang , 2006).

Las redes de Petri permiten modelar y estudiar sistemas concurrentes, asíncronos, distribuidos, paralelos, no deterministas y/o estocásticos. Como herramienta gráfica, dichas redes pueden utilizarse como un apoyo de comunicación visual similar a diagramas de flujo o diagramas de bloques, como una herramienta matemática, es posible establecer ecuaciones de estado, ecuaciones algebraicas y algunos otros modelos matemáticos que denoten el comportamiento del sistema modelado (Murata, 1989).

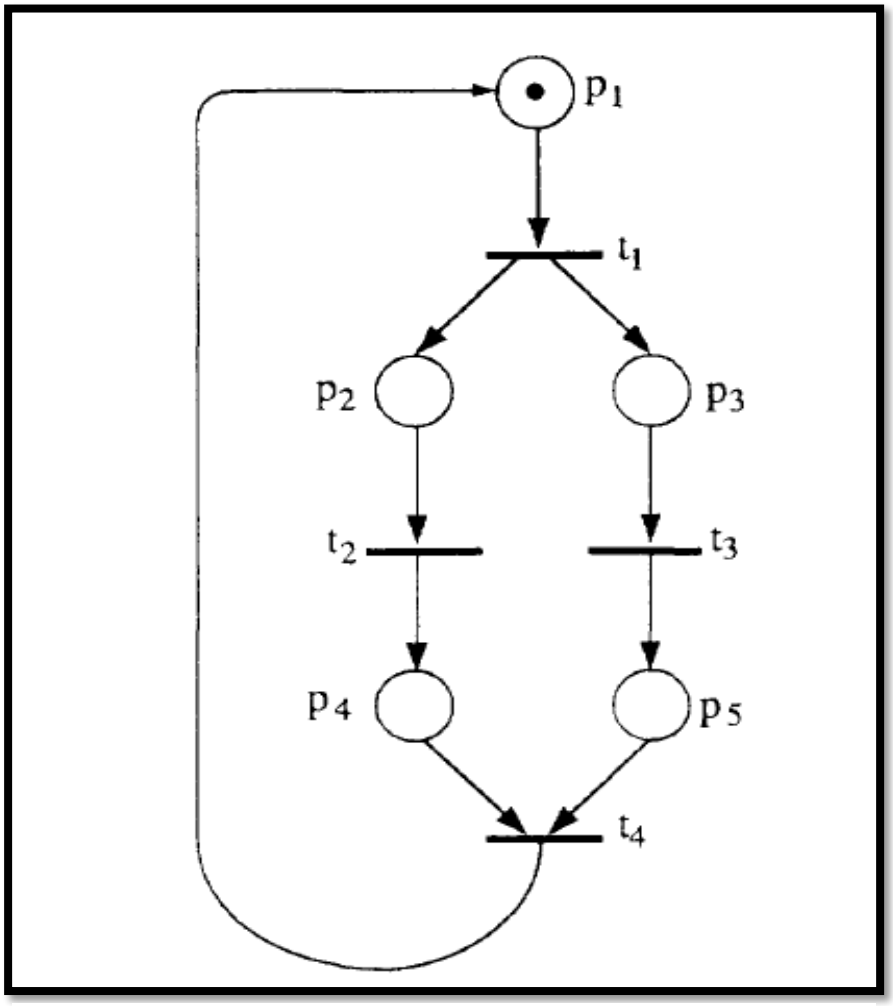

Figura 2. 1 Ejemplo de modelado de Redes de Petri, (Zurawski & Zhou, 1994).

2.1.2 Extensiones de Redes de Petri

Una red de Petri original que describe un sistema del mundo real tiende a ser compleja y muy grande por lo que no permite modelar características tales como cambios dinámicos, modos de operación múltiples, etc.

Es por ello que varios autores han diseñado extensiones del modelo básico de redes de Petri que incrementan poder de abstracción tomando en cuenta el tiempo o la aleatoriedad en la ocurrencia de eventos con el objetivo de garantizar un mayor desempeño de los sistemas (Llorens & Oliver, 2004).

a) Redes de Petri Temporizadas.

Se introdujeron para mejorar el rendimiento de los sistemas, este tipo de redes sirven para modelar sistemas donde se necesita evaluar la duración temporal de las transiciones y además son útiles para resolver conflictos de colisión, teniendo en cuenta tanto los cambios de estado como la duración del procesamiento de cada evento realizado.

Existen diferentes maneras de implementar el tiempo en una red de Petri por lo que este tipo de redes tienen otras variaciones que se pueden clasificar en:

• Redes de Petri temporizadas deterministas, donde la duración de los eventos es fijo (Granda, 2012).

•Redes de Petri temporizadas probabilísticas, donde el tiempo de duración de los eventos y la velocidad de disparo de las transiciones se especifican mediante funciones de distribución de probabilidad (Granda, 2012).

•Redes de Petri Estocásticas, se obtienen asociando con cada transición una variable aleatoria con distribución exponencial que exprese el retardo de la transición desde la habilitación hasta el disparo de la transición. La transición que tenga asociado el retardo más breve disparará primero (Lindemann, 1998).

b) Redes de Petri Coloreadas.

Utilizadas para modelar la lógica de un sistema, son similares a las redes de Petri estándar, la diferencia es que una red de Petri de color cada arco dirigido a las transiciones y cada token pueden llevar asociado un color que los diferencie de otros, además cada lugar y transición se le asigna un conjunto de colores y los colores de los tokens pueden cambiar en cada disparo de las transiciones, de acuerdo a la dinámica que se presenta en un sistema real (Liu , Blätke, & Heiner , 2014).

Las redes de Petri de color se utilizan para modelar el comportamiento de las células robóticas en la industria aeronáutica, simula gráficamente el comportamiento de las células en un entorno 3D, lo que permite la detección de colisiones y el cálculo de los tiempos de (Aguiar, Villani, & Junqueira, 2011).

c) Redes de Petri coloreadas Condicionales.

Es una extensión de redes de Petri Coloreadas fue desarrollada con la finalidad de definir reglas de tipo ECA (Evento-Condición-Acción), utilizadas en los Sistemas de Bases de Datos Activas. Al igual que una Coloreada, los tokens almacenan información correspondientes a los registros de las bases de datos, sin embargo, en esta red condicional se aplica una evaluación en la transición donde se almacena la parte condicional de la regla ECA (Medina Marin , 2002).

d) Redes de Petri Difusas.

Son utilizadas para representar las reglas de producción de un sistema experto que utiliza conceptos de lógica difusa, describen la relación entre dos proposiciones donde una transición habilitada se puede disparar sólo si el valor del token cumple con las características definido en la transición (Chen , Jyh-Sheng , & Chang, 2002).

e) Redes de Petri Generalizadas (RPG).

Son aquellas redes en las que se asigna un peso a los arcos y los arcos no etiquetados se les define valor de una unidad. La representación gráfica refleja el peso en número entero junto al arco (García Moreno, 1999).

f) Redes de Petri con Arcos Inhibidores.

En este tipo de redes un arco inhibidor que se representa por una línea que termina con un círculo, conecta una plaza a una transición. La condición de disparo de una transición se habilita cuando la cantidad de tokens de todas las plazas de entrada unidas a la transición con arcos normales, es igual al número de arcos normales y cuando no hay ningún token en las plazas de entrada unidas a las transiciones por arcos inhibidores (Huayna, Cortez Vásquez, & Vega Huerta, 2009).

g) Redes de Petri con Capacidad Limitada.

Son aquellas en las que cada plaza puede contener una capacidad limitada de tokens. Una transición estará habilitada sólo si en todos los lugares de entrada hay tokens y si el token que resulte de su disparo no viola las restricciones de capacidad establecidas. Un caso

particular de las redes de Petri limitadas son aquellas que poseen únicamente un token en la plaza, este tipo de redes reciben el nombre de redes de Petri (García Moreno, 1999).

2.1.3 Gráfico Funcional de Etapa-Transición (GRAFCET).

El modelo Grafcet fue definido por un grupo francés en 1977, es un derivado de redes de Petri binarias, es usado en el diseño de control automático basados en autómatas programables y en la lógica cableada basada en autómatas programables. Grafcet hereda la representación de una red de Petri, posee dos tipos de nodos, los pasos y las transiciones y propone una sola interpretación de las entradas-salidas, pero tiene unas diferencias, el marcado de un paso es Booleano y todas las transiciones se disparan simultáneamente, en la figura 2.2 se modela el llenado de un tanque utilizando GRAFCET, (René, 1995).

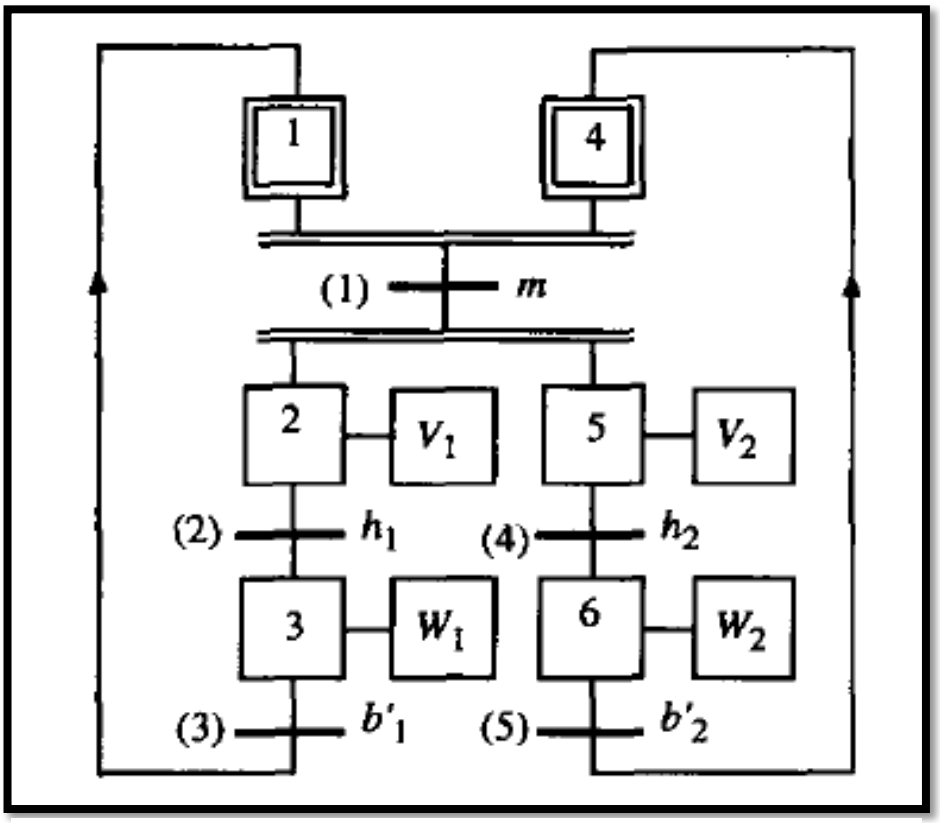

Figura 2. 2 Ejemplo de modelado GRAFCET en el llenado de un tanque, (René, 1995).

2.1.4 Gráficas De Funciones Secuenciales (SFC).

Se originó a partir de la notación Grafcet, que tiene sus raíces en las redes de Petri, un diagrama SFC es una notación gráfica para la descripción de los aspectos funcionales y de comportamiento de sistemas de control de eventos discretos. En SFC se representan los estados del sistema y las transiciones que pasan el control de un paso a uno o más pasos sucesores donde la duración de cada estado se determina por el disparo de las transiciones, las acciones que se ejecutan cuando su paso o estado asociado está activo y las condiciones de disparo (Jiang, Azzopardi, Holding, Carpenter, & J.S.Sagoo, 1996).

Los programas SFC suelen ser ejecutados por los controladores lógicos programables (PLCs) para tener un control de fabricación. Sin embargo, un obstáculo para el análisis de los programas SFC es que carecen de una descripción semántica completa, por lo tanto, es necesario traducir los modelos matemáticos de SFC a autómatas finitos o redes de Petri (Bauer, Huuck, Lukoschus, & Engell, 2004). En la figura 2.3 se representa una aplicación de SFC en el empacado de dos tipos de productos en tres estaciones de trabajo (Wightkin, Buy, & Darabi, 2011).

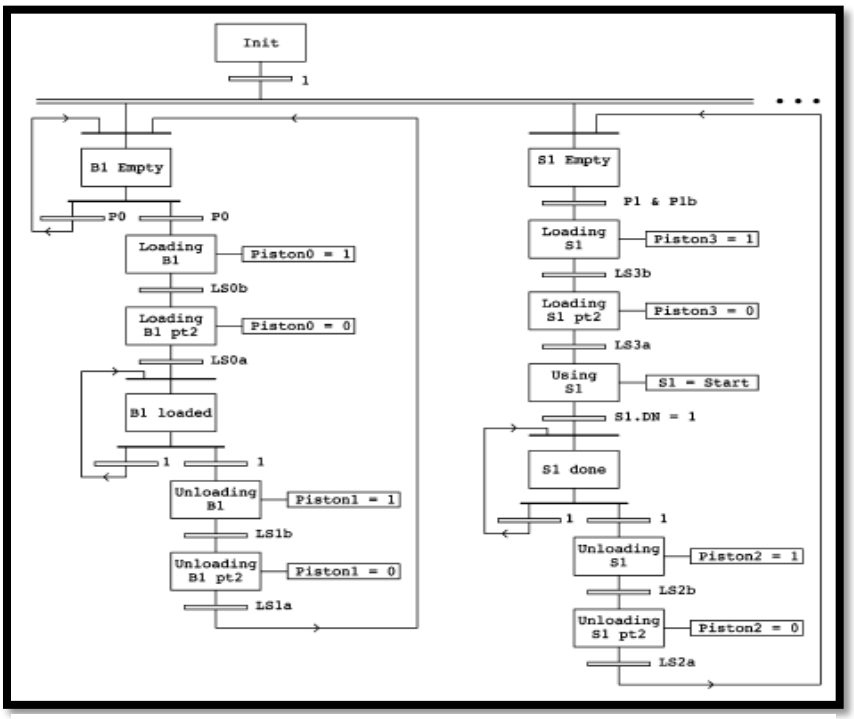

Figura 2. 3 Ejemplo de Modelado SFC en el empaquetado de dos productos, (Wightkin, Buy, & Darabi, 2011).

2.1.5 IDEF 0.

Esta técnica es conocida por el acrónimo en inglés de definición de la integración para modelado de las funciones (Integration Definition for Function Modeling), se originó como una técnica de ingeniería de software para la definición de los requerimientos del sistema, posteriormente se ha utilizado para describir las funciones de un sistema con el fin de examinar la forma en que se realizan las actividades; el modelo IDEF0 descompone un sistema en componentes más elementales más fáciles de tratar y va introduciendo gradualmente más niveles de detalle a través de la estructura del modelo. IDEF0, aporta una mejor comprensión del sistema a través de su representación gráfica, figura 2.4.

Un modelo IDEF0 consiste en un conjunto de diagramas jerárquicos que definen los límites del sistema y la estructura del sistema de una manera de arriba hacia abajo. Los diagramas consisten de cajas que definen funciones, actividades y tareas, flechas de datos que representan las relaciones o el flujo de información entre las funciones, controles que dictan cómo los insumos se transforman en salidas de la actividad y de mecanismos que representar a los recursos como personas, equipos, software, hardware, etc. responsables de la transformación (Haines & Evers, 1990). Actualmente existen nuevas versiones de IDEF hasta la versión 5.

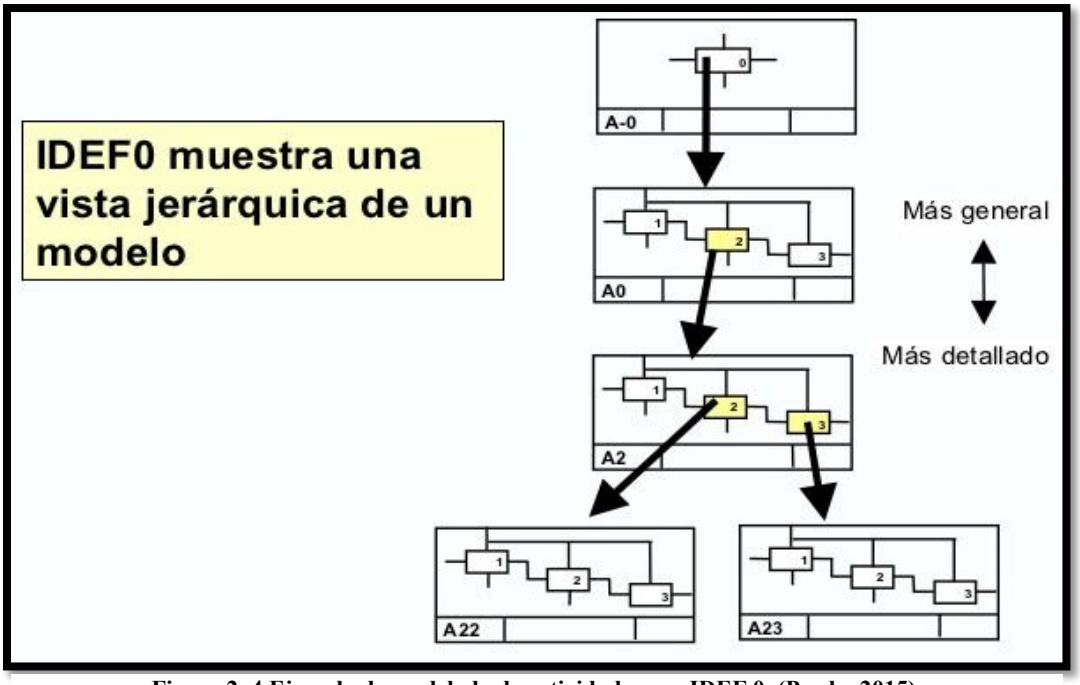

Figura 2. 4 Ejemplo de modelado de actividades con IDEF 0, (Prado, 2015).

2.1.6 Gráficas de Procedimiento de Funciones (PFC).

Sistema de modelado gráfico definido en el ANSI/ISA 88, similar al formato SFC. Tiene símbolos de inicio y final que indican los puntos de inicio y de finalización de un procedimiento. En la figura 2.5 se muestran los elementos de un diagrama PFC, donde cada rectángulo representa un elemento, una operación o una fase en el procedimiento de una receta, las pequeñas barras dobles bajo un paso indican una condición de transición con esto se señala que un equipo es llamado para realizar una operación, cada procedimiento espera hasta que el equipo termina la operación y entonces procede a realizar los siguientes pasos. Las transiciones entre elementos son opcionales y las operaciones de elementos en paralelo se representan por dos líneas horizontales (Dennis, 2006).

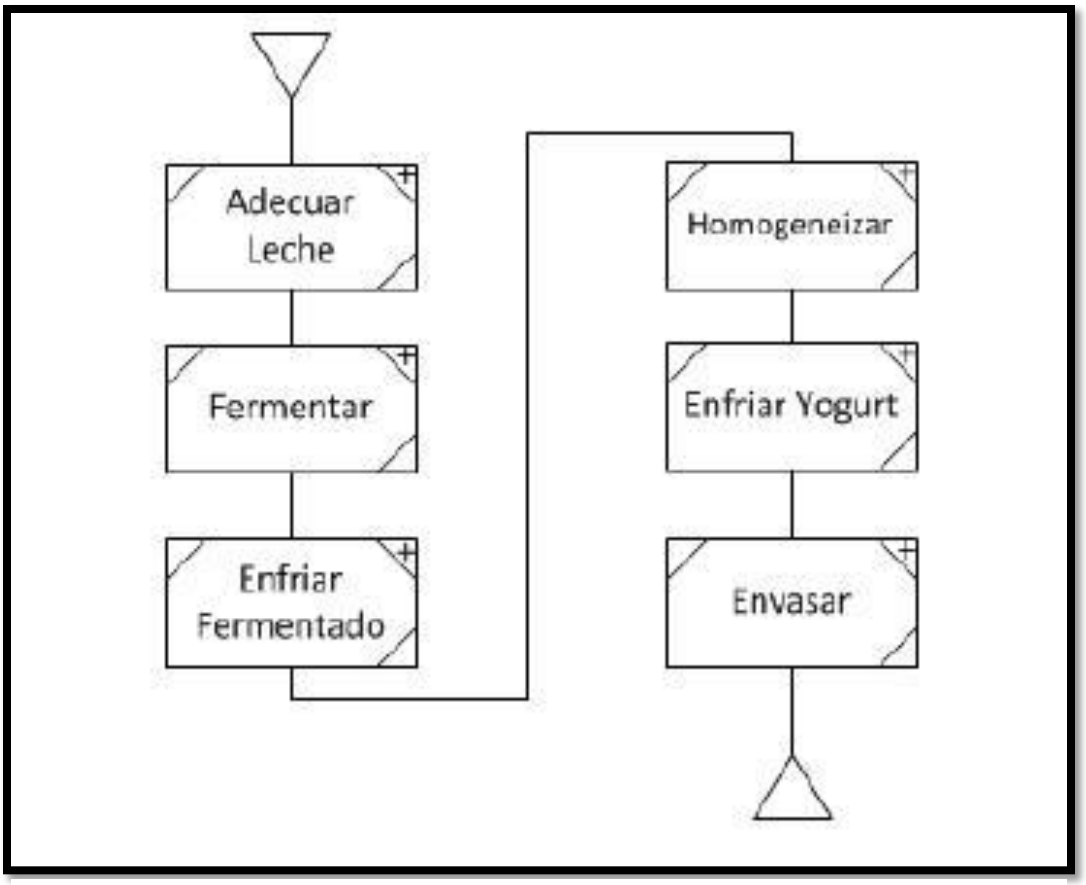

Figura 2. 5 Ejemplo de diagrama de procedimiento de funciones (PFC), (Dennis, 2006).

2.1.7 Lenguaje unificado de modelado (UML).

Lenguaje de Modelado Unificado, es un sistema de modelado para aplicaciones de software, sistemas hardware y aplicaciones del mundo real, en la figura 2.6 se muestran distintas maneras de modelar sistemas entre los cuales se encuentran diagramas de casos de uso para modelar procesos, diagramas de secuencia para modelar comunicación entre objetos, diagramas de estado para visualizar el comportamiento de los objetos en el sistema, etc.

Sin embargo el uso de UML para modelado de aplicaciones de automatización de hoy en día no es muy común. Una de las razones es la falta de semántica formal de UML, ya que provoca malentendidos entre los desarrolladores de las aplicaciones y esto genera problemática en proyectos de automatización bien programados, por lo que se ha creado una versión nueva, UML V2, que tiene más potencial para la automatización ya que cubre aspectos como distribución y concurrencia, así como recursos de automatización y las interfaces del dispositivo (Lange, Chaudron, & Muskens, 2006).

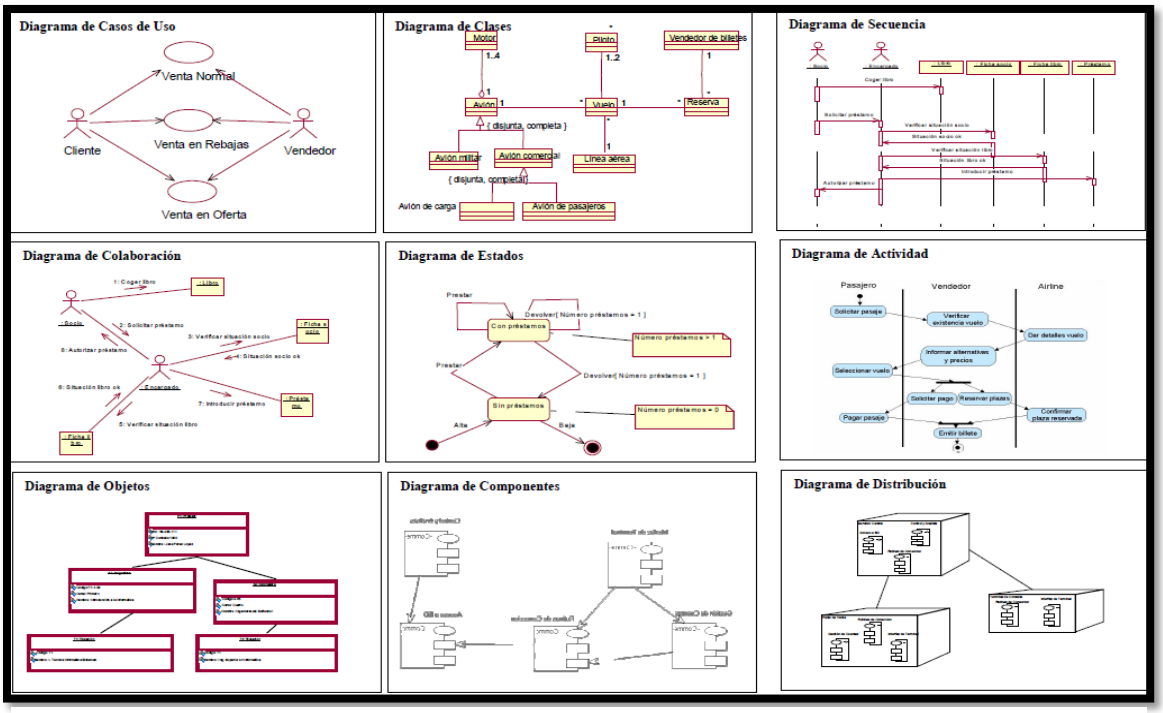

Figura 2. 6 Ejemplo de diagramas UML, (Eduardo, 2012).

2.1.8 Integrador Modular de Recursos de Producción (iMRP).

El Integrador modular de recursos de producción (iMRP) es una metodología de modelado para simplificar la comprensión, el diseño y la recopilación de información relevante de un piso de producción. El esquema iMRP, (Acosta & Sastrón F., 2007) está orientado a simplificar y mostrar de manera abstracta los procesos de fabricación, además se encarga de coordinar el flujo de las operaciones y el procesamiento de piezas con el equipo indicado extrayendo a simple vista la información requerida para ser interpretado por aplicaciones informáticas.

iMRP consiste en el análisis de características específicas y particularidades de un sistema de producción, esta información permite la construcción de un modelo gráfico, donde cada elemento del sistema dependiendo de sus atributos es representado por nodos y arcos. Los elementos se pueden clasificar en distintas clases como lo son: producto o pieza, equipo, operación y recurso tiempo. El conjunto de nodos o elementos se unen mediante arcos que indican la relación o la secuencia que existe entre los elementos. Por otra parte para simplificar aún más el modelado, en iMRP se agrupan la gran cantidad de operaciones que se realizan en el piso de producción en solo cuatro operaciones básicas, alimentación o retiro, transporte, procesamiento y manipulación.

El modelo integra los diferentes elementos y recursos (hardware y software) que componen a un sistema de producción en particular (piso de producción) y representa las diferentes etapas del ciclo de vida de un proceso en un mismo modelo particular. Los recursos que comprenden el piso de producción y la asociación entre los mismos se representan mediante dos elementos gráficos (nodos y arcos). En iMRP se propone un modelo orientado a objetos donde se modela de manera genérica un taller de manufactura en particular, donde la integración de los equipos se logra mediante el encapsulamiento de los mismos por clases del tipo genérica, con lo que el ambiente software componente se logra llevar a pie de máquina, reduciendo las particularidades de conexión por hardware entre aplicaciones de gestión y los equipos físicos (Acosta Cano de Los Rios, 2016).

En la figura 2.7 se representa el modelo iMRP, el cual es un grafo compuesto de nodos, donde destacan 4 nodos principales, el nodo producto representa información del conjunto de materiales requeridos (BOM) para la estructura del mismo, el nodo equipo representa la información sobre la manera en que se encuentra organizado el equipo (células,

estaciones de trabajo, etc.), el nodo operación agrupa a los posibles tipos de operaciones y el nodo recurso tiempo indica la secuencia de las operaciones.

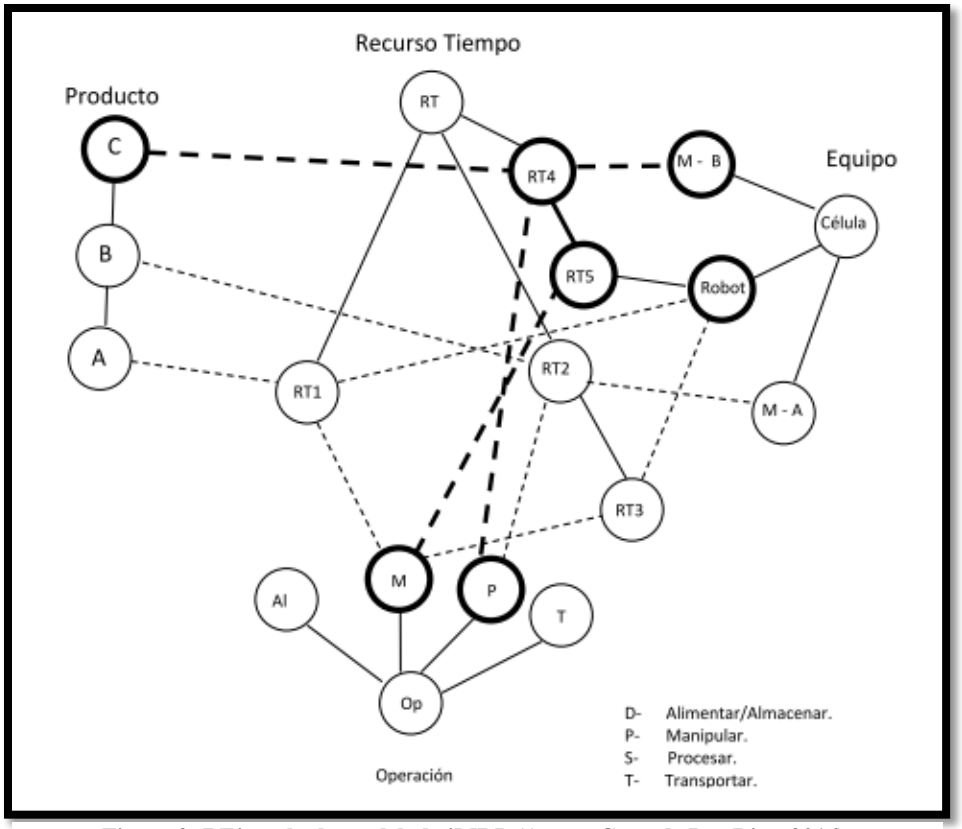

Figura 2. 7 Ejemplo de modelado iMRP (Acosta Cano de Los Rios, 2016).

2.2 OPERACIÓN DIRIGIDA POR MODELO.

El concepto de operación dirigida por modelo propone una relación entre un modelo de referencia y un sistema informático de control, dicha relación se da en el paradigma orientado a objetos entre una clase y una instancia de la misma, de esta forma los elementos del sistema de producción son representados en una relación uno a uno con objetos software; el sistema informático de control interpreta los elementos del modelo particular, que representan equipos, operaciones, elementos del producto, el orden (condiciones de precedencia) y la logística de las operaciones.

Una vez realizado el proceso de instanciación de los objetos del modelo particular, el sistema informático puede identificar la cantidad y el tipo de equipos existentes en el piso de producción y esto resulta de gran utilidad ya que el sistema de control crea objetos software

que representan a cada uno de los equipos físicos de producción. El sistema de control y los objetos equipo representantes operan en la misma plataforma de computación lo que se facilita la comunicación entre el sistema informático y el objeto representante y además es posible la incorporación de nuevos objetos representantes en tiempo de ejecución.

Después de que el sistema informático instancia a los objetos que representan a los equipos reales en el piso de producción, el sistema informático puede enviar ordenes de operación a los equipos (representantes) para que realicen alguna tarea específica tomando en cuenta la disponibilidad de los mismos y el orden de las operaciones interpretadas directamente del modelo particular.

2.3 ACOPLAMIENTO DÉBIL.

El acoplamiento débil se puede interpretar como una relación autónoma entre un sistema de control y los equipos de producción en un piso de fabricación, es decir el sistema es capaz de integrar nuevos equipos de producción sin un esfuerzo considerable y de establecer una comunicación flexible con ellos para enviarles órdenes de operación.

En (Acosta Cano de Los Rios, 2016) se plantea el concepto de acoplamiento débil, en donde se afirma que el concepto proviene del área de sistemas organizacionales, sin embargo, tomando como base esa estrategia de acoplamiento en un organigrama, se desarrolla un esquema de acoplamiento entre sistema informático y equipo de producción, en la tesis doctoral se presenta un detallado análisis de este concepto en donde se describen las condiciones que deben ocurrir para la existencia del acoplamiento débil, los tipos de acoplamiento débil y los compensadores a utilizar para soportar la operación del sistema bajo es el concepto de acoplamiento débil, introduciendo en el esquema planteado los componentes principales del acoplamiento (componentes tecnológico, tarea y rol).

Para lograr un flexibilidad, el sistema informático de control se debe ser capaz de integrar y de establecer una comunicación con nuevos equipos de producción sin un esfuerzo considerable de acoplamiento, esta capacidad que debe tener el sistema es un reto tomando en cuenta la gran variedad de equipos, fabricantes, modelos, protocolos de comunicación y lenguajes de programación que existen, además no existe un estándar que solucione todos los aspectos de integración y que sea aceptado por todos los fabricantes.

Con la finalidad de facilitar el análisis del problema de integración en la figura 2.8 se definió mediante una expresión compacta todos los elementos que intervienen en la comunicación entre el sistema informático y el equipo de producción lo cuales son: Función del medio físico (F1) y el fabricante de la circuitería (tarjeta) a utilizar (Pm), Conjunto de comandos (C) a ser ejecutados por el controlador del equipo, conjunto que depende del tipo (Te) y fabricante del equipo (Me); y el conjunto de argumentos (A) asociado con el comando correspondiente (C).

$$S\{F1(Pm), C(Te, Me), A(C)\} \quad (1)$$

Donde:

S = Sentencia de comunicación.

Pm = Medio físico de comunicación = |Ethernet|RS 232|RS 485|USB|GPIB|...

F1 = Función del medio físico = |SendData|WritePort|DataArrival|...

Te = Tipo de equipo de producción = |Robot|Torno|Horno|Prensa|Taladro|...

Me = Fabricante del equipo de producción = |Mitsubishi|Motoman|Bridgeport|Fagor|...

C = Comando = |Road|Reset|Mov|...

A = Argumento de comando = |Programa1|PosA|P1|...

Figura 2. 8 Formula de la sentencia de comunicación entre SI-EP (Acosta Cano de Los Rios, 2016).

Otro aspecto que se tomó en cuenta fue el efecto de operación del equipo el cual refleja el estado del equipo de producción y el estado de la pieza operada donde los estados dependen de los comandos ejecutados, en la figura 2.9 se muestra una expresión que describe los posibles efectos.

Ef {Es(E, C), Ps(P,O)} (2)

Donde:

Ef = Efecto de la operación.

E = Conjunto de equipos = |Eq1|Eq2|Eq3|...

C = Conjunto de comandos = |Encender|Operar|Pausar|Abortar|Mover|...

Es = Conjunto de estados de equipo = |Operando|Apagado|Cargado|Mantenimiento|...

P = Conjunto de piezas = |P1|P2|P3|...

O = Conjunto de operaciones = |Taladrar|Ranurar|Grabar|Pulir|Templar| |Mover|Cargar|...

Ps = Conjunto de estados de pieza = |Taladrada|Ranurada|Grabada|Pulida| |Templada| Pulida|...

Figura 2. 9 Formula del efecto de la operación del equipo (Acosta Cano de Los Rios, 2016).

En el análisis de integración (Acosta Cano de Los Rios, 2016) introduce los tres componentes del acoplamiento el rol, la tarea y tecnología, los cuales están implícitos en las expresiones ilustradas en las figuras 2.8 y 2.9, donde el rol que es definido como una abstracción de las posibles tareas a realizar por el equipo, la tarea que describe los detalles del proceso a realizar por el equipo y el componente tecnológico, que es la manera de comunicar la tarea a realizar por el equipo. En la Figura 2.10 se presenta de forma resumida los elementos que intervienen en la comunicación entre el sistema informático y el equipo de producción clasificados de acuerdo a los tres componentes de acoplamiento.

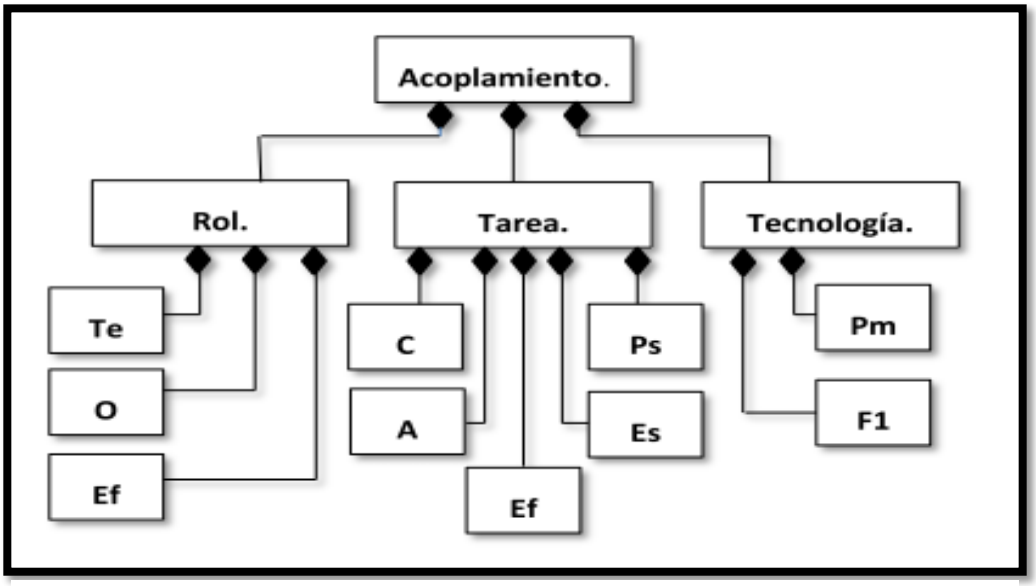

Figura 2. 10 Elementos de comunicación de los componentes de acoplamiento (Acosta Cano de Los Rios, 2016).

En su tesis doctoral, (Acosta Cano de Los Rios, 2016) menciona que para que el sistema informático de control sea flexible, describiendo esa flexibilidad como la facilidad de adaptarse a todos los posibles cambios que puedan ocurrir en un piso de producción sin tener que modificar el código fuente del sistema es necesario que exista un acoplamiento débil entre los módulos del sistema.

En la tabla 1 se describen los elementos y características del acoplamiento débil en el área organizacional y su adaptación al dominio de Sistema Informático y Equipo de Producción.

Tabla 1. Aspectos relevantes del modelo propuesto por Orton y Weick interpretados en el dominio de integración SI-EP, (Acosta Cano de Los Ríos, 2016).

Modelo de Orton y Weick		**Traducción al Dominio SI-EP**
Voces del Modelo	**Elementos de Acoplamiento**	
Ambiente de un sistema débilmente acoplado.	Varios medios conducen al mismo fin	Existen diferentes alternativas para lograr el mismo fin; amplia variedad de equipos con la capacidad de realizar el mismo proceso.
	Relativa ausencia de regulaciones	Existencia y aceptación de una amplia variedad de estándares desde diferentes puntos de vista tales como naturaleza, fabricante o modelo de equipo.
Tipos de acoplamiento débil.	Acoplamiento entre niveles jerárquicos.	Acoplamiento entre los niveles sistema informático y controlador del equipo
	Acoplamiento entre intenciones y acciones.	Acoplamiento entre la intención de realizar cierto proceso y tanto el equipo disponible para ello como la libertad en la toma de decisiones para lograrlo.
Compensadores.	Liderazgo ampliado	Comandar discretamente y delegar los detalles de implementación al nivel equipo.
	Enfoque de atención	Centrarse en los aspectos esenciales para el control del equipo.
	Valores compartidos	Acoplamiento utilizando preferiblemente, tecnologías de propósito general.
	Persistencia	Sistema informático (SI) debe permanecer invariable ante cambios en los tipos de equipo, lenguajes, estándares de operación y plataforma de computación.

Consecuencias del acoplamiento débil.	Amortiguamiento (Buffer)	Modificaciones en el conjunto de equipos de producción (EP) no deben afectar al sistema informático (SI).
	Adaptabilidad	Asimilación y adaptación al cambio
	Experimentación	Adaptabilidad que soporta identificación de soluciones
	Juicio Colectivo	Uso de estándares de propósito general en lugar de estándares de un dominio particular, (valores compartidos).
	Discrepancia	Incluya situaciones no tomadas en tiempo de diseño
	Satisfacción	Facilidad para cubrir necesidades (menor conexión entre elementos)
	Efectividad	Logro de mejores soluciones a través de experimentación e innovación.

Además en la tabla 2 se describe la relación entre los compensadores, los componentes de acoplamiento débil y los elementos de integración de equipo y operación.

Tabla 2.- Resumen del esquema de referencia para la integración SI-EP basado en componentes de acoplamiento y compensadores de acoplamiento débil, (Acosta Cano de Los Rios, 2016).

	Valores Compartidos	**Enfoque de Atención**	**Liderazgo Ampliado**	
			Discreción	**Delegación**
Rol (Te,O,Ef)	Alentar el uso de conceptos y tecnología ampliamente aceptados y disponibles.	Monitoreo del efecto sobre la pieza de la operación del equipo (basado en el rol del equipo)	Cuatro tipos de equipos: Procesador Manipulador Transporte Alimentación/Retiro	Conjunto de objetos delegados atendiendo los aspectos relacionados con el rol.
Tarea (C,A,PS,Es, Ef)		Centrarse en las principales etapas del proceso (ejemplo: inicio y fin de operación)	Encapsulamiento de la tarea en un documento.	Objetos representantes para actualizar el estado de la

		Identificación de la pieza y equipo operándola.	Especificación de la tarea mediante nombre y ubicación del documento.	pieza y del equipo. Manejo de la comunicación de la tarea a través de un objeto envoltura.
Tecnología **(Pm,F1)**		Ubicación de los actores (objetos) en ambos extremos del acoplamiento (dirección de la cola o red) Ubicación de archivos de tarea.	Incorporación dinámica de código de clases de objeto delegado y propulsor (Selección en tiempo de ejecución)	Uso de objetos delegados envoltura y propulsor para manejar sintaxis y medio físico del controlador del equipo.

Con base en el análisis realizado sobre diferentes aspectos del acoplamiento débil, en el esquema propuesto por (Acosta Cano de Los Rios, 2016) las expresiones de integración entre el sistema informático y el equipo de producción se reducen los elementos del conjunto de variables para cada aspecto del problema de integración sistema informático/equipo de producción. En la tabla 3 se muestra una relación entre las posibles variables de integración y las variables que maneja el sistema informático bajo los conceptos de acoplamiento débil.

Tabla 3.- Conjuntos de variables en la Integración SI-EP, (Acosta Cano de Los Rios, 2016).

Elemento de integración	Posibles variables en el ambiente	Variables a manejar por el sistema informático
Pm=Medio físico de comunicación	|Ethernet|RS232|RS245| USB|GPIB|…	Conjunto vacío
F1= Función del medio físico	|SendData|WritePort|DataArrival|…	|Operar|Pausar|Detener| Abortar|

Te= Tipo de equipo de producción	|Robot|Torno|Horno |Prensa|Taladro|…	|Procesador|Manipulador|Transporte |Almacenamiento/Retiro
C= Comandos	|Encender|Operar| |Pausar|Abortar| Mover| |Road|Reset|…	|Operar|Pausar| Detener| Abortar|
A= Argumentos	|Programa1|PosA|…	Nombre del Archivo y ubicación
Es= Conjuntos de estados de Equipo	|Operando|Apagando| |Cargando| |Mantenimiento|…	|Vacante|Operando|Terminado|Pausa|Pausando|Detenido|Deteniendo|Abortado|Abortando|
O= Conjunto de Operaciones	|Taladrar|Ranurar| Grabar|Pulir|Templar|Mover|Cargar|…	|Procesamiento|Manipulacion|Transporte|Almacenamiento/Retiro|
Ps= Conjunto de estados de pieza	|Taladrada|Ranurada| Grabada|Pulida| Templada|…	|Procesada|Manipulada|Transportada|Alimentada/ Removida|

2.4 ARQUITAM.

El proceso de desarrollo de un sistema de fabricación inicia por la definición de los requerimientos del sistema, para continuar con su diseño e implementación, y posteriormente su operación y actualización. Sin embargo, para cada una de estas etapas se requieren abstraer o modelar las características del sistema necesarias para su realización. Existen diversas arquitecturas y modelos de referencia que cubren todas las etapas del ciclo de vida de un sistema de fabricación, sin embargo, podría afirmarse que en general no soportan la manera de realizar la transición entre una fase y la siguiente y resultan complejas de aplicar.

ArquiTAM (Arquitectura de referencia de Taller Automático de Manufactura), propuesta en (Acosta, C. & Sastron, 2006) consiste en una estructura de referencia, orientada a objetos, para pisos de producción desarrollada en el Instituto Tecnológico de Chihuahua en conjunto con la Universidad Politécnica de Madrid en donde se plantea un esquema de modelado como soporte para el desarrollo de un sistema de fabricación discreta, en particular, para el área de ejecución y control de producción.

La arquitectura de referencia ArquiTAM, está orientada a conducir la realización de las etapas del ciclo de vida del taller de manufactura, con la finalidad de lograr un sistema de fabricación con operación dirigida por modelo, con alto nivel de integración tanto entre las fases del ciclo de vida (para facilitar su desarrollo) así como entre los elementos que lo componen (para facilitar su operación y mantenimiento).

ArquiTAM presenta la organización del sistema de producción en diferentes niveles de abstracción directamente relacionados, que corresponden al nivel conceptual (Arquitectura de Referencia), la implementación genérica (Modelo de referencia genérico) y la implementación particular (Modelo particular) tal como se presenta en la figura 2.11.

El primer nivel comprende al universo de conceptos con base en las características comunes extraídas de los sistemas de fabricación discreta existentes, requerimientos y conceptos solución. El segundo nivel corresponde al desarrollo de un modelo de referencia genérico construido con base en las herramientas del marco de trabajo, en este nivel se delimita el alcance que tendrá el sistema partiendo de la definición de actividades a realizar y de los requerimientos generales con que debe cumplir un sistema de fabricación discreta. El tercer nivel es la implementación del sistema en casos particulares, para lo cual se adapta, mediante instanciación, el sistema genérico a las particularidades del entorno destino sin cambiar su estructura principal.

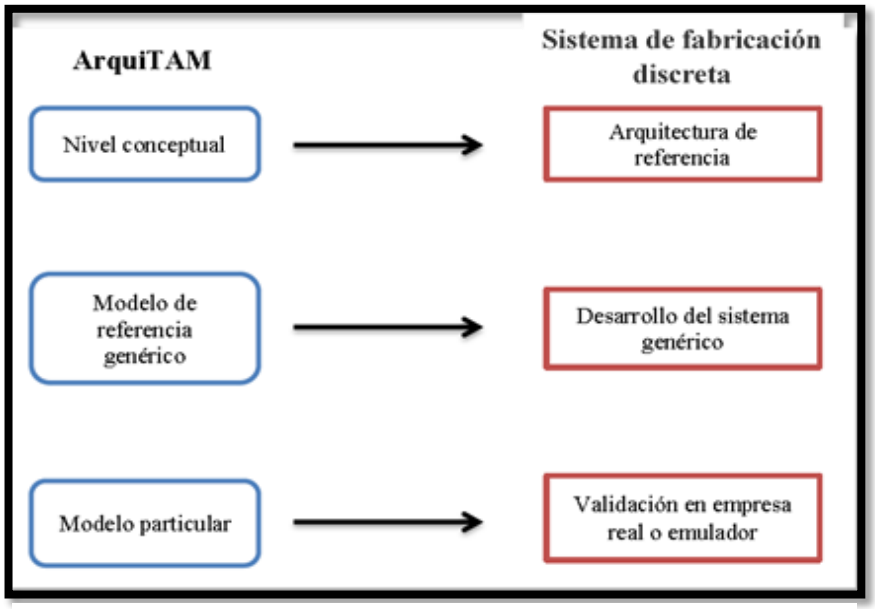

Figura 2. 11 Niveles de la arquitectura de referencia ArquiTAM, (Gómez García, 2012).

a) Arquitectura de Referencia (Nivel Conceptual)

En el nivel arquitectura de referencia (aspecto conceptual) se definen los elementos comunes del área. Una vez identificados estos elementos se diseña un modelo que pueda servir como punto de partida para el desarrollo de sistemas de fabricación discreta, correspondiente al siguiente nivel de la estructura propuesta en ArquiTAM. Partiendo de este modelo genérico es posible la implementación de un caso particular, a través de la abstracción de particularidades respecto a los elementos de hardware y software y los procesos de fabricación; abstracción que se realiza en forma de modelo interpretable por el sistema informático de control (sistema genérico de control), empleando así el concepto de operación dirigida por modelo.

b) Modelo de referencia genérico

El nivel modelo de referencia planteado en ArquiTAM consiste en la selección de herramientas para la implementación de un sistema genérico, con base en los conceptos solución identificados en el nivel de arquitectura de referencia. La flexibilidad del sistema radica en la posibilidad de poder adaptarse a cualquier tipo de procesos en un entorno de producción, por lo tanto, para desarrollar los módulos del sistema genérico es importante seleccionar herramientas y componentes que se acoplen fácilmente entre ellos para obtener una total compatibilidad al momento de la integración del sistema (plataforma Windows). Es importante destacar la relación entre arquitectura de referencia y modelo de referencia (o sistema genérico) establecida en ArquiTAM, donde es posible desarrollar diferentes modelos de referencia a partir de una misma arquitectura de referencia (conceptos).

c) Modelo Particular

ArquiTAM cuenta con un módulo para el manejo de particularidades en sistemas informáticos de gestión de producción llamado integrador modular de recursos de producción (iMRP), el cual ha sido aplicado a sistemas de coordinación de flujo y procesamiento de pieza en sistemas de fabricación discreta, (Acosta & Sastrón F., 2007).

A grandes rasgos la forma de aplicar ArquiTAM sería contemplar un conjunto de herramientas de software y hardware destinadas al desarrollo e implementación de sistemas

informáticos en el área de automatización industrial. En la figura 2.12 se hace un análisis de las actividades del taller y cómo se relacionan entre sí para obtener un modelo de referencia de taller. Por último se definen los requerimientos particulares del taller y se realiza la implementación del sistema informático que auxiliará en la automatización del taller. En la Siguiente figura se muestra de manera gráfica el concepto de ArquiTAM.

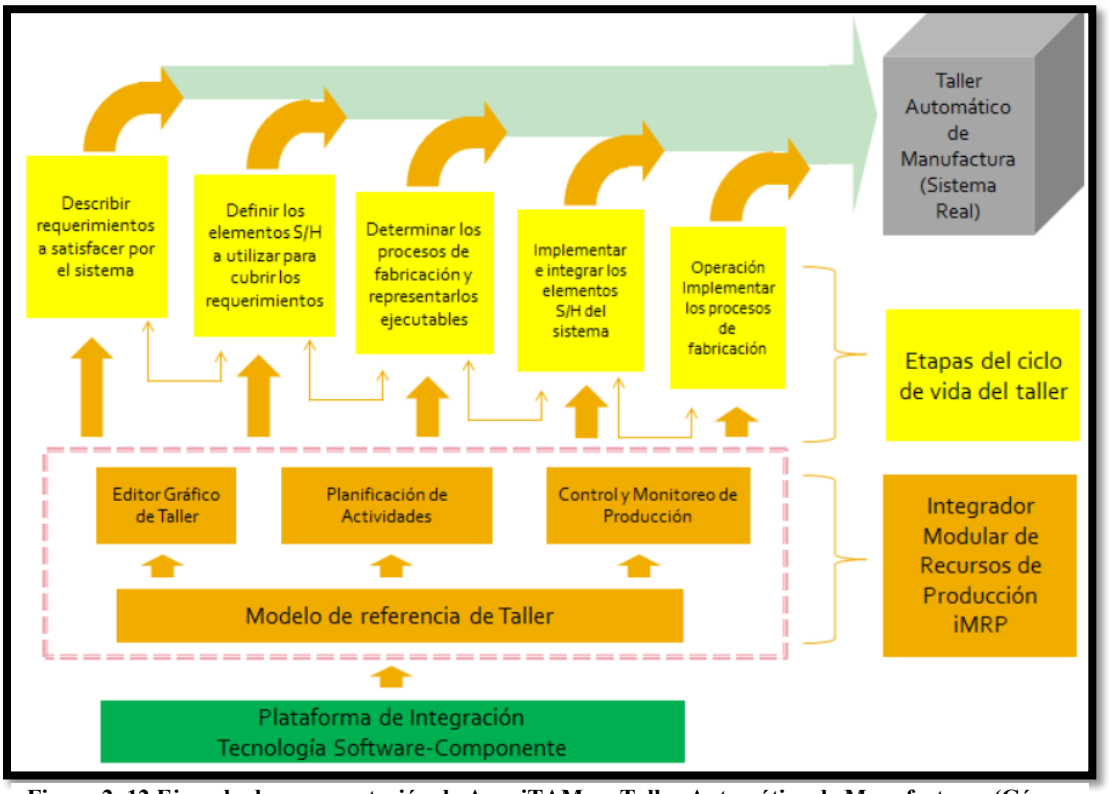

Figura 2. 12 Ejemplo de representación de ArquiTAM en Taller Automático de Manufactura, (Gómez García, 2012).

Una de las ventajas de ArquiTAM es que auxilia en el diseño de sistemas para sistemas automáticos de manufactura sin importar el giro de los mismos gracias a la flexibilidad que propone, otro aspecto relevante a tomar en cuenta consiste en la integración de las etapas de desarrollo del sistema como es el paso entre las etapas de definición de requerimientos, diseño, implementación y operación del mismo.

Al aplicar las recomendaciones de la estructura de referencia ArquiTAM para desarrollar un sistema informáticos, se logra el desarrollo de un sistema genérico capaz de adaptarse a distintos entornos. Al diseñar una arquitectura de referencia sirve como punto de partida para la construcción de un modelo genérico para sistemas de fabricación discreta que

pueda ser adaptable y flexible con el fin de aplicarse en un entorno particular sin la necesidad de modificar sus características estructurales.

Por otra parte, al desarrollar un sistema genérico dividido en módulos se logra un acoplamiento débil lo que hace más robusto en la medida de que si un módulo falla los otros pueden seguir operando casi en su totalidad. La flexibilidad obtenida en el sistema genérico se debe en gran parte a las herramientas y tecnologías seleccionadas para su desarrollo. Trabajar bajo una misma plataforma de desarrollo es de gran ayuda para la construcción de un sistema modular, ya que todos los componentes que funcionan bajo esta plataforma de alguna manera son compatibles entre sí. Por otra parte al crear un sistema bajo las características de esta arquitectura, este mostrará una gran flexibilidad, vista ésta como la facilidad para reconfigurar el sistema. La facilidad de realizar cambios de programa a ejecutar por los equipos sin modificar el código del sistema principal, indica en cierto sentido la flexibilidad del sistema. Otra perspectiva de flexibilidad es la correspondiente a nivel elemento, como pudiera ser la facilidad para realizar cambios en cuanto al equipo físico que forma a la estación, como sustituir equipo por otra versión, modelo o fabricante diferente o bien agregar nuevos equipos al piso de producción.

2.5 EFECTIVIDAD GLOBAL DEL EQUIPO (OEE).

El concepto de Efectividad Global del Equipo acrónimo en inglés de Overall Equipment Effectiveness (OEE) fue originado en Japón en 1971 cuando el Instituto Japonés de Mantenimiento de Plantas promovió el Mantenimiento Productivo Total (TPM) acrónimo en inglés de (Total Productive Maintenance) en el cual se incluyó el OEE, sin embargo OEE comenzó a ser reconocido a finales de los años 80’s cuando Seiichi Nikajima fundador del TPM introdujo estos conceptos a América. (Sheu, 2006).

OEE es un indicador que muestra el desempeño de un equipo de producción mediante el porcentaje de utilización real efectiva del mismo con respecto a su situación ideal equivalente a operación en óptimas condiciones, donde la diferencia la constituyen las pérdidas de tiempo, las pérdidas de velocidad y las pérdidas de calidad.

2.5.1 Objetivos del OEE.

Para optimizar los procesos productivos en las empresas de manufactura es importante monitorear el rendimiento de los equipos y las líneas que operan en el piso de producción para así poder cuantificar la productividad y eficiencia de los mismos.

En el sector manufacturero se ha utilizado la técnica OEE por más de diez años para medir la eficiencia de los equipos de fabricación ya que comúnmente la producción diaria del equipo se queda muy por debajo de la capacidad para el que fue diseñado. El objetivo de aplicar esta técnica de medición es medir los principales parámetros fundamentales del desempeño del equipo en un piso de producción industrial que son: la disponibilidad, el desempeño y la calidad, los cuales generan información que permite determinar con gran precisión como se están utilizando los recursos en el piso de fabricación y refleja cuáles son las causas que generan más perdidas, lo cual ayuda a gestionar acciones para abordar los problemas que ocurren en los procesos, a mejorar la eficiencia y la capacidad, bajar los costes, mejorar las ganancias, así como también permite estimar la necesidad de personal, materiales, equipos para poder lograr mantener una producción con los niveles más exigentes a nivel mundial, (Hansen, 2001).

2.5.2 Beneficios de OEE.

OEE es una herramienta de medición de mejora continua que no sólo permite conocer el rendimiento productivo de la maquinaria industrial sino que también permite medir el rendimiento de las líneas de producción. (Singh, Shah, Gohil, & Shah, 2012).

(Ames, Gililland, Konopka, & Schnabl, 1995) Muestran algunos beneficios de implementar OEE en el grupo SEMATECH entre los que se destacan facilidad para dar mantenimiento a los equipos, tasa de mejor calidad por la reducción de los efectos de un fallo del equipo, se reducen los costes, el tiempo y la frecuencia de reparación de equipos y aumenta el tiempo productivo de los equipos, entre otros.

2.5.3 OEE y TPM.

El mantenimiento predictivo total (acrónimo en inglés TPM) es una forma sistemática de trabajo para crear un proceso de producción sin pérdidas, al menor costo posible y con la

participación de todos los empleados mientras que OEE analiza y califica los diferentes tipos de pérdidas (esta clasificación proviene del TPM, en el que se definen las "Seis Grandes Pérdidas") que pueden producirse en un proceso productivo las cuales reducen el tiempo efectivo de proceso y la producción óptima a alcanzar (Pomorski, 1997). OEE es el producto de seis pérdidas de equipos agrupados en tres categorías, disponibilidad, eficiencia en el rendimiento y la tasa de calidad (Singh, Shah, Gohil, & Shah, 2012).

OEE	SEIS GRANDES PERDIDAS RECOMENTADAS	SEIS GRANDES PERDIDAS TRADICIONALES
PERDIDA DE DISPONIBILIDAD	Paradas no planeadas	Fallas del Equipo
	Paradas planificadas	Arranque y Ajustes
PERDIDA DE EFICIENCIA	Pequeñas paradas	Pequeñas paradas y puestas en marcha
	Ciclos lentos	Velocidad reducida
PERDIDAS DE CALIDAD	Rechazos de producción	Proceso defectuoso
	Rechazos de inicio	Producción reducida

Figura 2. 13 Seis grandes perdidas

Disminución de Disponibilidad

Son pérdidas de tiempo durante el cual el equipo debería haber estado produciendo pero no lo ha estado y por lo tanto ningún producto sale de la máquina.

1) Falla en el equipo (Paradas no planificadas): Es cualquier período de tiempo significativo en el que está previsto el equipo para la producción, pero no se está ejecutando por un fallo de algún tipo. La causa de esta disfunción puede ser técnica u organizativa, puede ser por error al operar la máquina, falta de herramientas o por mantenimiento pobre del equipo. El OEE considera este tipo de pérdida a partir del momento en el cual la avería aparece.

2) Configuración y Ajustes (Paradas Planificadas): Es cualquier período de tiempo significativo en el que el equipo no está funcionando debido a un cambio o ajuste, ya sea por limpieza de los equipos, tiempo de calentamiento, mantenimiento planificado, por inspecciones de calidad o por un paro para ir a almorzar o un cambio de turno, o cambio de herramientas. El OEE considera todo el tiempo en el cual la máquina no fabrica ningún producto.

Disminución de Rendimiento

Son pérdidas de velocidad lo cual implica que la máquina está funcionando pero no a su velocidad máxima.

3) Puesta en Marcha y Pequeñas paradas (Micro paradas): Representa el tiempo en que el equipo se detiene por un corto período de tiempo (menos de cinco minutos) y no trabaja a velocidad constante, generalmente son causadas por problemas de alimentación, atascos de material, el flujo del producto obstruida, una configuración incorrecta, sensores mal alineados o bloqueados, las cuestiones de diseño del equipo, y una rápida limpieza periódica. Estos pequeños problemas disminuyen de forma drástica la efectividad de la máquina, sin embargo, generalmente no se registran como una pérdida de tiempo.

4) Velocidad Reducida (Ciclos Lentos): Es la diferencia entre la velocidad fijada en la actualidad y la velocidad máxima de diseño. En muchos casos, la velocidad de producción se disminuye para evitar otras pérdidas tales como defectos de calidad y averías, o bien, equipo sucio o desgastado, mala lubricación, materiales de calidad inferior, malas condiciones ambientales, inexperiencia del operador, el arranque y la parada. Las pérdidas debidas a velocidades reducidas son en la mayoría de los casos ignoradas o infravaloradas.

Pérdidas de Calidad

Ocurre cuando la máquina fabrica productos que no son buenos a la primera. Se pueden diferenciar dos tipos de pérdidas de calidad:

5) Defectos de Proceso (Scrap): Son aquellos productos que no cumplen los requisitos establecidos por calidad, el objetivo es fabricar siempre productos de primera calidad desde la primera vez. Las causas más comunes de pérdida de calidad se dan en los arranques ya que la producción no es estable inicialmente, o bien, por mala configuración del equipo o errores del operador al manejar los equipos.

6) Rendimiento Reducido (Rechazos de arranques): Son los productos defectuosos producidos desde el arranque hasta que la producción estable se alcanza. El rendimiento reducido puede ocurrir después de cualquier puesta en marcha de equipos, sin embargo, es más comúnmente seguido después de los cambios por ajustes incorrectos cuando se ejecuta una pieza nueva.

2.5.4 Medición del OEE.

La medición estándar conocida como OEE debe ser exacta, en tiempo real y debe diagnosticar las causas por la que una línea no es totalmente efectiva. La medición de la efectividad del equipo se realiza a través de un porcentaje, que es calculado combinando tres elementos asociados con cualquier proceso de producción:

- Disponibilidad: tiempo real en que la máquina está produciendo.
- Rendimiento: la producción real de la máquina en un determinado periodo de tiempo.
- Calidad: Producción sin defectos.

En (Oechsner, Pfeffer, & Pfitzner, 2003) se muestra la captura y clasificación de las pérdidas de tiempo disponible de los equipos, así como también se analizan los datos de las pérdidas para dar la prioridad a acciones correctivas.

Además en la figura 2.14 se señala que la productividad del equipo se ve afectada en gran medida por factores mucho más allá de los equipos, incluyendo operador, receta, instalaciones, materiales, disponibilidad, requisitos de programación, etc.

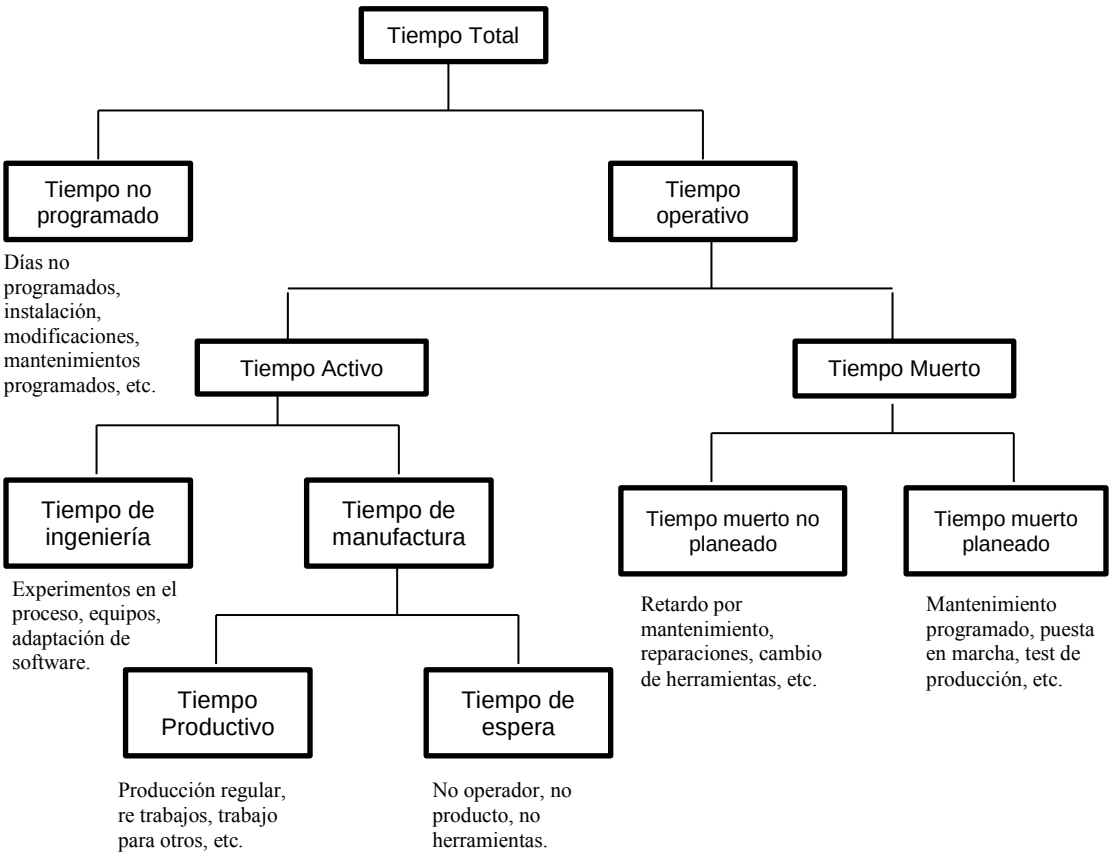

Figura 2. 14 Elementos que afectan la productividad del equipo, (Oechsner, Pfeffer, & Pfitzner, 2003).

2.5.5 Calculo del OEE.

(Hansen, 2001)Indica que existen distintas maneras de calcular el métrico OEE y señala que todas llegan al mismo resultado, además describe las formulas básicas planteadas por S. Nikajima en la "Introducción a TPM". En la figura 15 se identifican los elementos principales para obtener el OEE y (Vorne, 2016) explica cada una de estas partes detalladamente.

Figura 2. 15 Representación de las pérdidas que permiten calcular el OEE, (Vorne, 2016).

OEE es el resultado de multiplicar la Disponibilidad del Equipo por la Eficiencia por la Calidad, (Singh, Shah, Gohil, & Shah, 2012) describe el procedimiento para calcular este métrico, en (Vorne, 2016) se analiza cada elemento de la fórmula.

Disponibilidad

Disponibilidad = (Tiempo Operado)/ (Tiempo Neto Disponible) [1]

Donde:

Tiempo Neto Disponible=Tiempo total programado-Paradas Planeadas.

Tiempo Operado=Tiempo Neto Disponible-Paradas no planeadas.

Eficiencia

Eficiencia= (Total de Piezas Producidas)/ (Tiempo Operado x Vel Max del Equipo) [2]

Calidad

Calidad= (Total Piezas Producidas-Piezas de mala calidad)/ (Total Piezas Producidas) [3]

OEE

OEE=Disponibilidad x Eficiencia x Calidad [4]

2.5.6 Clasificación del OEE.

El valor del OEE permite clasificar una o más líneas de producción, o toda una planta, con respecto a las mejores de su clase y que ya han alcanzado el nivel de excelencia. De esta manera se tiene la siguiente clasificación: (Hansen, 2001).

a) OEE < 65% Inaceptable. Se producen importantes pérdidas económicas. Muy baja competitividad.

b) 65% < OEE < 75% Regular. Aceptable sólo si se está en proceso de mejora. Pérdidas económicas. Baja competitividad.

c) 75% < OEE < 85% Aceptable. Continuar la mejora para superar el 85 % y avanzar hacia niveles de Clase mundial. Ligeras pérdidas económicas. Competitividad ligeramente baja.

d) 85% < OEE < 95% Buena. Entra en niveles de Clase mundial. Buena competitividad.

e) OEE > 95% Excelencia, niveles de Clase mundial. Excelente competitividad.

CAPÍTULO 3. PLANTEAMIENTO DEL PROBLEMA

3.1 INTRODUCCIÓN.

Las empresas de manufactura discreta integran tecnología para automatizar sus procesos de fabricación utilizando diversos dispositivos tales como PLC´s (Programable Logic Controller), sensores, actuadores, instrumentos de medición e instrumentos robóticos que realizan operaciones programadas repetidamente en las células de trabajo, con el fin de lograr un flujo continuo de producción. Al mismo tiempo implementan sistemas informáticos para la administración de la planta que les permite gestionar procesos y controlar la calidad de los mismos en un piso de producción. Sin embargo, debido a las exigencias tan variantes del mercado muchas veces es necesario hacer cambios de equipos, de procesos o de productos lo cual genera esfuerzo tecnológico y pérdidas económicas a las empresas. Debido a esto surge la necesidad de implementar un sistema de control flexible a nivel de piso de producción que sea capaz de cambiar o incorporar nuevos equipos de producción y nuevos productos sin la necesidad de invertir una considerable cantidad de recursos en reconfigurarse.

Crear un sistema informático de control de producción capaz de integrar nuevos equipos de producción sin un esfuerzo considerable de acoplamiento representa grandes retos tomando en cuenta la gran variedad de equipos, fabricantes, modelos, protocolos de comunicación y plataformas de programación que existen, además, las empresas de manufactura discreta tienden a manufacturar una gran variedad de productos lo que las obliga a reconfigurar sus líneas de fabricación para satisfacer las nuevas necesidades de producción, invirtiendo así grandes cantidades de tiempo y esfuerzo al modificar sus sistemas informáticos para poder acoplar los nuevos equipos de fabricación.

No existe una manera estándar y universal de desarrollar un sistema informático de control, por lo que la mayoría de los desarrollos existentes carecen de un orden y son llevados a cabo sobre la marcha.

En (Acosta Cano de Los Rios, 2016) se propone una solución para crear un sistema informático de control flexible capaz de integrar nuevos equipos y de comunicarse con ellos sin importar el medio de comunicación, utilizando la técnica de modelado iMRP y un esquema de acoplamiento débil. Tal solución se basa en el uso de la arquitectura de referencia ArquiTAM para crear un sistema informático modular donde los componentes de hardware

y software son separados claramente, buscando la flexibilidad que permita la modificación de alguno de los módulos sin afectar al resto (Acosta & Sastrón F., 2007).

En ArquiTAM (Arquitectura de referencia de Taller Automático de Manufactura) se presenta un esquema de tres niveles para lograr crear un sistema informático de control flexible, en el primer nivel (Arquitectura de referencia) se plantean los requerimientos y las posibles soluciones para facilitar el desarrollo del sistema durante la etapa de implementación (en nuestro caso los requerimientos radican en crear un sistema de control capaz de adaptarse a cambios de equipos y de productos; por otra parte establecer comunicación entre el sistema de control y los equipos de producción; como solución se propone crear un sistema modular donde cada módulo se acople débilmente con los otros, es decir, que los cambios de un módulo sean aislados al resto del sistema. El segundo nivel de la arquitectura (modelo de referencia) consiste en implementar las soluciones planteadas en el nivel anterior, mediante la creación de una herramienta informática genérica donde se aplican métodos de desarrollo de software para crear en forma de código componentes genéricos (clases reusables) que faciliten la instanciación del sistema para una aplicación en particular. En el tercer nivel (modelo particular) consiste en instanciar los componentes genéricos (clases) para un caso en particular, es decir, a partir de las clases genéricas definidas en el nivel anterior se crean objetos software que representan los recursos reales existentes en un piso de producción específico. Además, en la arquitectura ArquiTAM se plantea la técnica de operación con base en un modelado gráfico iMRP en donde se abstraen los elementos del piso de fabricación y las características específicas de los equipos de producción, y se describen los procesos de fabricación de manera ejecutable por aplicaciones informáticas.

Las empresas no sólo buscan la integración flexible de nuevos equipos, también necesitan monitorear las actividades que se llevan a cabo en sus procesos para determinar con mayor precisión como se utilizan los recursos y equipos en el piso de fabricación y así detectar las causas que generan las pérdidas importantes, lo cual ayuda a gestionar acciones para abordar los problemas que ocurren en los procesos, a mejorar la eficiencia, bajar los costos, mejorar las ganancias, entre otras.

Una manera de medir la eficiencia de los procesos de producción en una empresa es utilizando el métrico de desempeño conocido como Efectividad Total del Equipo, OEE (acrónimo en inglés de Overall Equipment Effectiveness), el cual mide los parámetros

fundamentales en la producción industrial que son: la disponibilidad, el desempeño y la calidad; además informa cuales son las principales perdidas en los procesos.

En el presente trabajo de investigación se presenta una implementación de un sistema informático de control lanzador de órdenes al piso de producción, basado en los conceptos de acoplamiento débil, operación dirigida por modelo y la arquitectura de referencia ArquiTAM, haciendo uso de la plataforma de desarrollo Visual Basic .NET para la creación de los módulos que componen el sistema. Para la comunicación entre los objetos representante y envoltura se hace uso de los servicios de cola de mensajes de Windows (MSMQ). Por otra parte, en el módulo envoltura (de equipo) se integran funciones para calcular el desempeño de los equipos (OEE) y para generar reportes que indican las principales causas de tiempos muertos en los equipos.

3.2 OBJETIVOS.

3.2.1 Objetivo General.

Desarrollar una herramienta informática flexible para el control y monitoreo en sistemas de fabricación discreta con base en el concepto de operación dirigida por modelo (iMRP).

3.2.2 Objetivos Específicos.

1. Identificar los elementos conceptuales y operacionales de los tres niveles de la arquitectura del esquema ArquiTAM para el control de un sistema de producción discreta.

2. Desarrollar un sistema informático para modelar, controlar y simular una planta de producción discreta con base en la técnica de modelado iMRP y el concepto de acoplamiento débil.

3. Desarrollar el sistema de control de estaciones de trabajo emuladas por robots tipo Robobuilder.

4. Calcular el métrico de desempeño OEE (Overall Equipment Effectiveness) en los equipos de producción.

3.3 JUSTIFICACIÓN.

Las plantas de fabricación discreta se componen de complejas células de trabajo y de múltiples líneas de producción en las cuales se realizan una gran cantidad de operaciones automatizadas para ensamblar componentes hasta obtener un producto final terminado. Por tal motivo, es importante que las empresas que utilizan procesos de producción discreta tengan un sistema de modelado que les permita observar fácilmente los procesos en un piso de fabricación.

iMRP es una técnica de modelado que abstrae las particularidades de un piso de producción, técnica comprendida en el esquema de referencia ArquiTAM, que ofrece grandes ventajas para modelar un sistema a nivel de piso de producción en forma interpretable por sistema informático. Este modelo es fácil de representar y puede ser analizado y comprendido fácilmente por los ingenieros de la empresa. Así mismo, resulta interpretable por un sistema informático, situación que se pretende aprovechar en el presente proyecto de tesis para la aplicación del concepto de operación dirigida por modelo. De esta manera, iMRP será la base para el diseño y creación de un sistema informático de interés académico que permita modelar la secuencia, los procedimientos y recursos necesarios para la fabricación de productos con base en método de operación dirigida por modelo.

Calcular el OEE ofrece una ventaja competitiva a las empresas ya que permite detectar las causas que generan pérdidas significativas asociadas a los equipos de producción lo cual apoya en la toma de decisiones para evitar más pérdidas.

3.4 APORTACIÓN PRÁCTICA.

La utilidad de este proyecto está definida en los siguientes puntos:

- Existe un interés académico en el área de operación dirigida por modelo, concepto que facilitaría la adaptación (flexibilidad) al sistema informático de control a cambios en el sistema de producción. Así mismo, uno de los retos en esta dirección es la facilidad para la generación del modelo a realizar por el personal de manufactura sin requerir conocimientos de computación.

- Utilizando los conceptos de operación dirigida por modelo (técnica de modelado iMRP) en el modelado del sistema de producción, se busca obtener una manera práctica de identificar procesos y recursos en el piso de producción.

- Además de la operación dirigida por modelo, en el presente proyecto de tesis se busca aplicar el esquema ArquiTAM para aportar flexibilidad al sistema informático de control de producción respecto a su integración con el conjunto de equipos de producción.

- Calcular el métrico de desempeño (OEE) será de gran utilidad para medir la efectividad de los equipos en un piso de producción discreta y a partir de esto se podrán detectar las causas potenciales que provocan su mal rendimiento lo cual ayudará a realizar acciones preventivas y correctivas para solucionar los problemas que ocurren en los procesos.

CAPÍTULO 4. DESARROLLO DE LA SOLUCIÓN PROPUESTA

En este capítulo se describe el desarrollo del sistema propuesto como solución al problema planteado en el capítulo anterior. El sistema informático de control lanzador de órdenes, se plantea en una estructura modular que incluye una aplicación capaz de calcular métrico de desempeño OEE y de generar reportes con información relevante de los equipos del piso de producción.

4.1 SISTEMA INFORMÁTICO DE CONTROL.

En la figura 4.1 se presenta un diagrama de bloques para representar cada uno de los módulos que conforman el sistema informático de control.

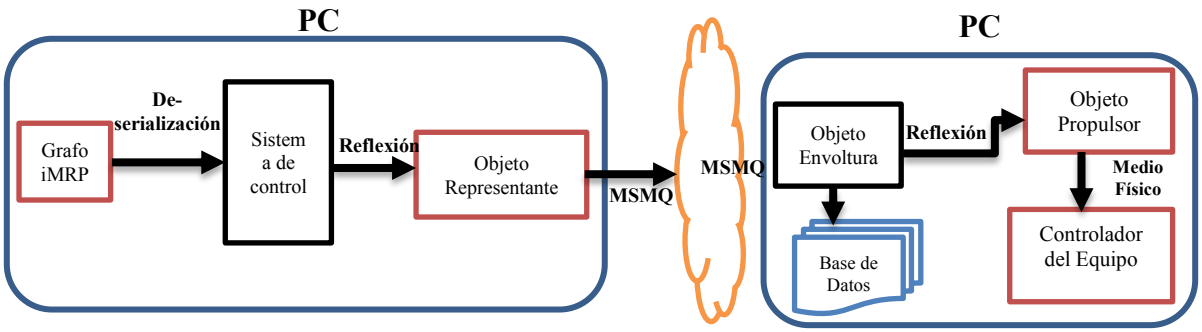

Figura 4. 1 Diagrama a bloques del Sistema Informático de Control

El primer módulo es el sistema de control el cual se encarga de agregar dinámicamente a los objetos representantes e interpretar modelos iMRP (abstracción de los elementos del piso de producción tales como equipos, BOM del producto y flujo de operaciones), los cuales están serializados en formato XML; con base en la información del modelo envía órdenes de operación a los objetos representantes de equipo.

El siguiente modulo consiste de objetos representantes, que representan de forma virtual a los equipos existentes en el piso de producción y se encargan de transmitir las órdenes de operación provenientes del sistema de control al objeto envoltura utilizando el servicio de cola de mensajes de Windows (MSMQ); este módulo (objetos representantes) se encuentra en la misma computadora y está diseñado en la misma plataforma de desarrollo que el sistema de control (Visual Basic) con la finalidad de simplificar la comunicación entre ellos. Los objetos representante son creados dinámicamente por el sistema de control con base en el modelo iMRP utilizando la técnica de reflexión que ofrece el framework de .NET.

El objeto envoltura es otro modulo que compone al sistema que generalmente está en una computadora distinta a los objetos representantes por lo que la comunicación entre ellos se da mediante un protocolo independiente a la plataforma (MSMQ, Servicios Web, entre otros). El objeto envoltura es el responsable de recibir y procesar las órdenes de operación provenientes de los objetos representantes y se encarga de transmitir dichas operaciones al objeto propulsor. Este módulo se encarga de agregar dinámicamente a los objetos propulsores(driver) mediante la técnica de reflexión tal y como lo hace el sistema de control con los objetos representantes, además el objeto envoltura está conectado a una base de datos SQL Server en la cual se registran la hora y fecha cada vez que se recibe o se envía un mensaje al objeto representante, esto con la finalidad de calcular el tiempo que operan los equipos y los tiempos muertos que ocurren y así poder aplicar técnicas de medición para obtener la eficiencia real de los equipos en el piso de producción.

El objeto propulsor (driver) representa al equipo físico de manera virtual y es el encargado enviar dichas órdenes directamente al controlador del dispositivo físico, en él se establecen los comandos para poder manejar el medio físico de comunicación con el que cuenta el controlador del dispositivo. El lanzador de órdenes y el objeto propulsor se encuentran implementados en la misma plataforma computacional, de la misma manera que el objeto envoltura y objeto driver, con la intención de facilitar la incorporación de código en tiempo de ejecución, lo que es posible mediante el uso de librerías dinámicas aplicando la técnica de reflexión.

4.1.1 Modelo de Particularidades (iMRP).

Para la aplicación de la técnica de modelado iMRP (Acosta & Sastrón F., 2007) se empleó el editor gráfico (González Carrasco, 2017). Cada modelo gráfico consta de arcos y nodos, existen 4 tipos de nodos los cuales son clasificados en nodo producto, nodo operación, nodo equipo y nodo recurso tiempo como se muestra en la figura 4.2.

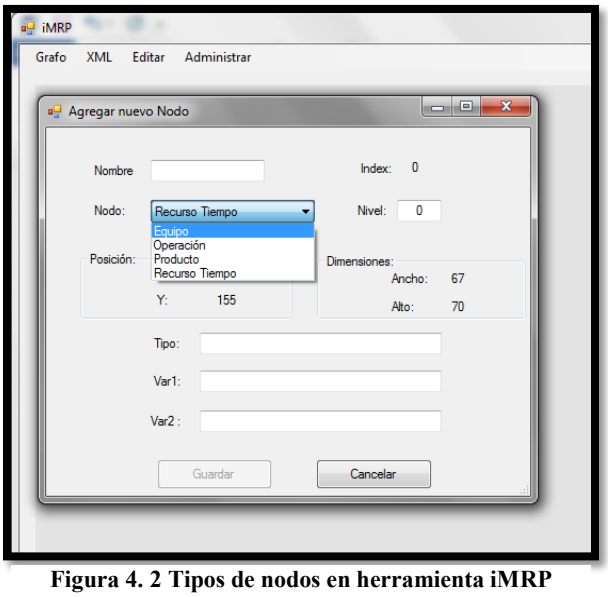

Figura 4. 2 Tipos de nodos en herramienta iMRP

El nodo equipo contiene información relevante en sus propiedades, ya que guarda información sobre el tipo de equipo, el nombre y la dirección por donde envía (var1) y recibe mensajes (var2) a través del servicio de cola de mensajes de Windows MSMQ. Figura 4.3

Por otra parte, el arco que une al nodo operación y al nodo recurso tiempo contiene la dirección del archivo de texto en el cual se encuentran las instrucciones para que el equipo físico realice determinadas operaciones. Figura 4.4

Figura 4. 4 Propiedades del nodo equipo.

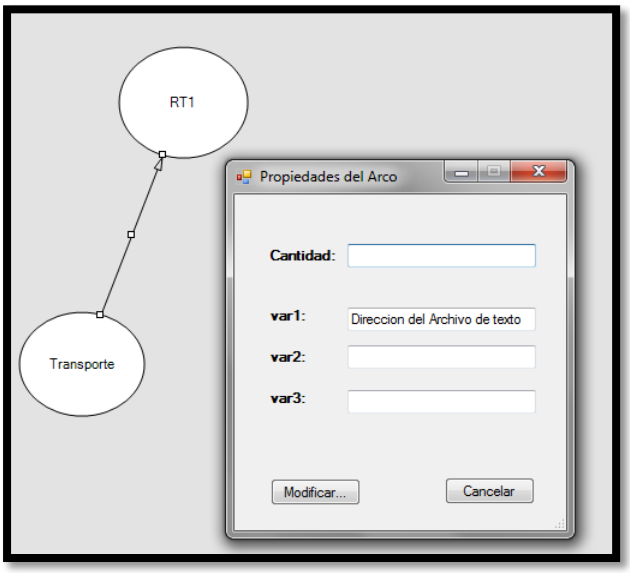

Figura 4. 3 Propiedades del Arco que une el nodo operación con el nodo Recurso Tiempo

El modelo iMRP creado para una situación particular se guarda en formato XML para su interpretación por el sistema de control, figura 4.5.

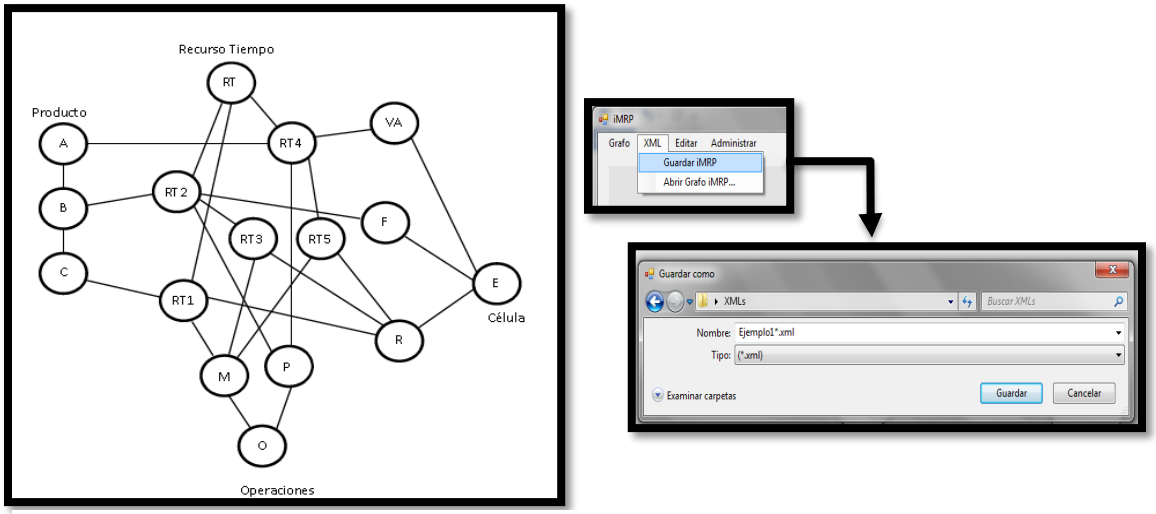

Figura 4. 5 Guardar grafo iMRP en archivo de tipo XML

4.1.2 Módulo 1: Sistema Informático de Control.

El sistema informático de control es una aplicación con una interfaz gráfica desarrollada en la plataforma de Visual Basic .NET, es el módulo principal del sistema, encargado de interpretar modelos iMRP los cuales contienen de manera abstracta la información relevante del piso de fabricación (Equipos, piezas, el BOM y el flujo de operaciones, etc.), además, es responsable de crear instancias de los objetos representantes necesarios y de enviar ordenes de operación de manera ordenada a los equipos del sistema de producción.

4.1.2.1 Clases Genéricas.

La primera función del sistema de control es interpretar el modelo de particularidades iMRP, para ello es necesario crear algunas clases genéricas en la plataforma de desarrollo Visual Basic las cuales contienen las mismas propiedades que el modelo gráfico, las siguientes clases deben de ser generadas para poder de-serializar el modelo iMRP ya que en cada una de ellas se guarda información relevante del modelo.

- Clase Grafo. Esta clase está compuesta por el nombre del grafo y una lista de arcos y nodos.
- Clase Arco. Aquí se encuentran variables como: origen y destino para enlazar dos nodos, además de las variables cantidad, var1 la cual se utiliza para guardar la ruta del archivo de texto cuando existe una relación origen-destino entre un nodo operación y un nodo recurso tiempo y las variables var2 y var3 para incluir información extra.
- Clase Nodo. En esta case hay variables en las cuales se guarda la información del nodo tales como:
    - Nombre. Aquí se guarda el nombre del nodo
    - Tipo. Puede ser tipo equipo, producto, operación o recurso tiempo
    - TipoObj. En caso de ser tipo equipo en esta variable se guarda la ruta donde está localizada la DLL del equipo.
    - Var1. En caso de ser tipo equipo en esta variable se guarda la ruta de la cola de mensajes por donde se recibirá mensajes el equipo.
    - Var 2. En caso de ser tipo equipo en esta variable se guarda la ruta de la cola de mensajes por donde se enviará mensajes el equipo.
- Clase Operación. En esta clase están contenidas las variables Id la cual sirve para identificar el origen o destino, Nombre y Archivo en la cual se guarda la ruta del archivo de texto
- Clase Producto. Cuenta con las siguientes variables:
    - Id: Variable que indica el id del nodo el cual sirve para identificar el origen o destino
    - Nombre. Se guarda el nombre de la pieza
    - Estado. Variable para indicar el estado de la pieza (procesada o no procesada)
    - Padre. Variable para indicar cuál es el nodo principal
    - Hijos. Contiene una lista de nodos precedentes
    - Destino_Producto. Variable booleana que indica si un nodo de tipo producto es destino de otro nodo producto
- Clase Recurso Tiempo. En esta clase se agrupa la información de varias clases y está conformada por las variables:

- Id. Variable que indica el id del nodo el cual sirve para identificar el origen o destino
- Nombre. indica el nombre de la variable recurso tiempo
- Producto. Variable de tipo producto que sirve para relacionar al objeto producto con el recurso tiempo
- Operación. Variable de tipo operación que sirve para relacionar al objeto operación con el recurso tiempo
- Equipo. Variable de tipo objeto que sirve para relacionar al objeto equipo con el recurso tiempo
- RTs_Precedentes. Es una lista en la cual se guardan los objetos recurso tiempo precedentes
- DestinoRT. Variable booleana que indica si un nodo de tipo recurso tiempo es destino de otro nodo del mismo tipo.
- Terminado. Variable de tipo booleana que indica el estado del objeto recurso tiempo (terminado, en espera).

La finalidad de estas clases genéricas es guardar la información del modelo grafico iMRP en cada una de sus variables, para así poder interpretar la información y lograr enviar órdenes de operación en los equipos.

La información del modelo iMRP está contenida en un archivo XML, sin embargo para que el módulo de control pueda interpretar dicha información es necesario de-serializar el archivo y guardar toda la información del mismo en las clases mencionadas anteriormente.

4.1.2.2 Archivos XML.

Al iniciar la operación del sistema de control (lanzador de órdenes) se debe seleccionar el archivo XML que contiene los elementos del modelo iMRP tales como equipos, operaciones, piezas y Recursos Tiempo, luego, el sistema debe deserializar el archivo y crear una instancia de la clase genérica "grafo" para guardar la información del modelo grafico en las listas de arcos y nodos de la clase tal como se muestra en el listado 4.1.

Listado 4. 1 Método para des serializar archivo XML

```
Public Sub DeserializarXML(ByVal Path As String)
        Try
            Dim ArchivoXML As New StreamReader(Path)
            Dim y As New XmlSerializer(GetType(Grafo))
            obGrafo = y.Deserialize(ArchivoXML)
            ArchivoXML.Close()
            MsgBox("Archivo Deserializado Exitosamente")
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try
    End Sub
```

La información del modelo iMRP contenida en las listas de nodos y arcos de la clase grafo, se emplea en crear diferentes instancias de objetos y guardarlos en listas de objetos utilizando las clases genéricas dependiendo del tipo de nodo tal como se muestra en el listado 4.2. Una vez creados los diferentes tipos de objetos, ahora se deben de relacionar entre sí para poder obtener el orden en el que los equipos realizaran las operaciones, para ello a los objetos Recurso Tiempo se les asigna una operación, una pieza y un equipo y con base en las relaciones de los arcos en el modelo iMRP se determina el orden en que se ejecutaran los objetos recurso tiempo tomando en cuenta la disponibilidad del equipo y la fase de la pieza que se esté procesando.

Listado 4. 2 Método donde se crean y se guardan en listas los objetos de tipo Producto, Operación, Recurso tiempo y Equipo

```
Public Sub CrearObjetos()
    For Each nodo As Nodo In obGrafo.Nodos
            If nodo.Tipo.Equals("Recurso Tiempo") Then
                    obRT = New RT(nodo.Nombre, nodo.Index)
                     listaRT.Add(obRT)
            End If
            If nodo.Tipo.Equals("Producto") Then
                    obProd = New Producto(nodo.Nombre, nodo.Index)
                    listaProd.Add(obProd)
           End If
           If nodo.Tipo.Equals("Operación") Then
                    obOperacion = New Operacion(nodo.Nombre,
nodo.Index, "")
                    listaOper.Add(obOperacion)
           End If
           If nodo.Tipo.Equals("Equipo") Then
                    reflexion(nodo.TipoObj, nodo.Nombre, False,
nodo.Index, nodo.Var1, nodo.Var2)
            End If
    Next
     For Each arco As Arco In obGrafo.Arcos
     obArco = New Arco()
     obArco.Origen = arco.Origen
     obArco.Destino = arco.Destino
     obArco.Var1 = arco.Var1
     listaArcos.Add(obArco)
   Next
End Sub
```

4.1.2.3 Instancias de Objetos Representante.

Cuando el sistema de control interpreta el archivo XML (modelo iMRP) y detecta un nodo de tipo “Equipo”, crea instancias de los objetos representantes los cuales son Librerías de clases (DLL´s) que se agregan dinámicamente por reflexión al sistema de control en tiempo de ejecución, esto brinda un mayor grado de flexibilidad a la aplicación ya que se pueden agregar distintos equipos según lo indique el modelo. Para poder utilizar las propiedades de reflexión es necesario incluir en el proyecto de sistema de control la referencia de System.Reflection, la cual permite hacer uso de la clase Assembly y Type.

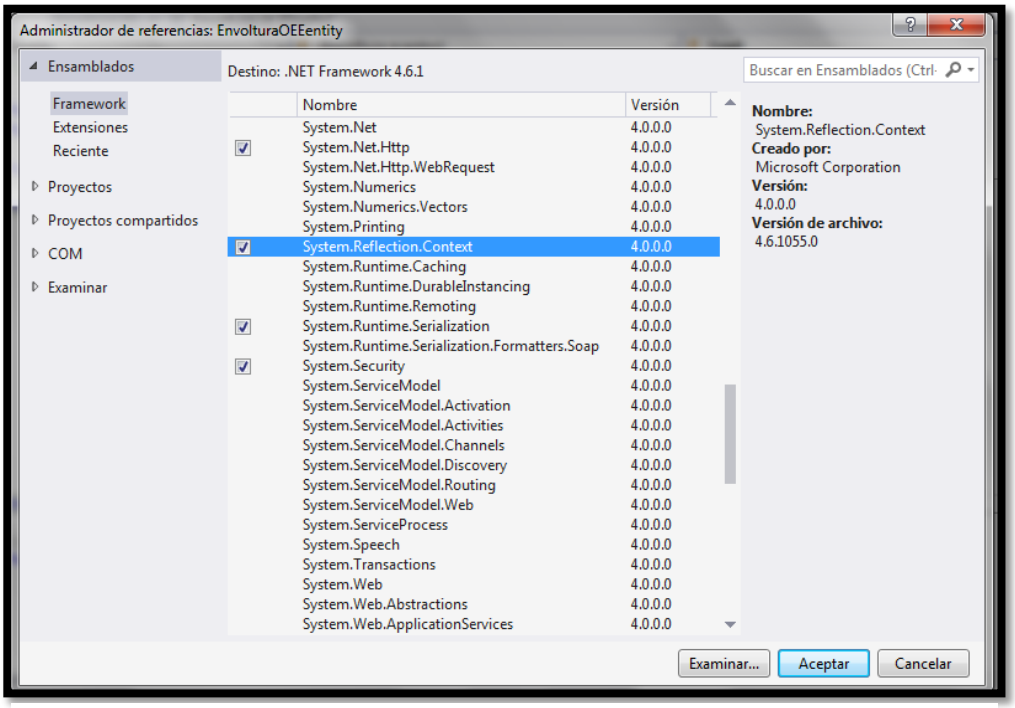

Figura 4. 6 Agregar Referencia System.Reflection al proyecto Sistema de Control

En el listado 4.3 se puede observar la incorporación del objeto Assembly (librería dinámica a incorporar) a partir de la ruta del archivo .dll y la obtención de su tipo para posteriormente crear instancia.

Listado 4. 3 Incorporación dinámica de librería por reflexión

```
Dim TestArray() As String = Split(direccionDLL, "\\")
    If TestArray.Length > 1 Then
       dll = TestArray(0)
       clase = TestArray(1)
       objAsm = Assembly.LoadFile(dll)
       objEqReflexion = objAsm.CreateInstance(clase)
    End If
```

La instrucción Assembly.LoadFile (dll) indica que se desea cargar el archivo seleccionado a una variable de tipo Assembly, posteriormente con la instrucción objEqReflexion = objAsm.CreateInstance (clase) se obtiene el tipo de la clase a utilizar para crear una instancia. Una vez que se realizan estas instrucciones se obtiene la instancia de la clase equipo representante y se vuelve posible utilizar sus métodos y propiedades.

4.1.3 Módulo 2: Objeto Representante.

En esta sección se presenta el desarrollo del objeto representante, para ello se utilizó la plataforma de visual Basic 2015 con lo cual se logró desarrollar un proyecto de tipo librería de clases en el cual se incluyen métodos y propiedades para poder llevar a cabo la comunicación con el objeto envoltura.

Esta entidad del sistema da como resultado un archivo .dll el cual es incluido en la aplicación de control aplicando técnicas de reflexión (incorporación dinámica de código).

4.1.3.1 Librerías de Clases (DLL).

Para crear el objeto representante es necesario crear un nuevo proyecto del tipo librería de clases, figura 4.7.

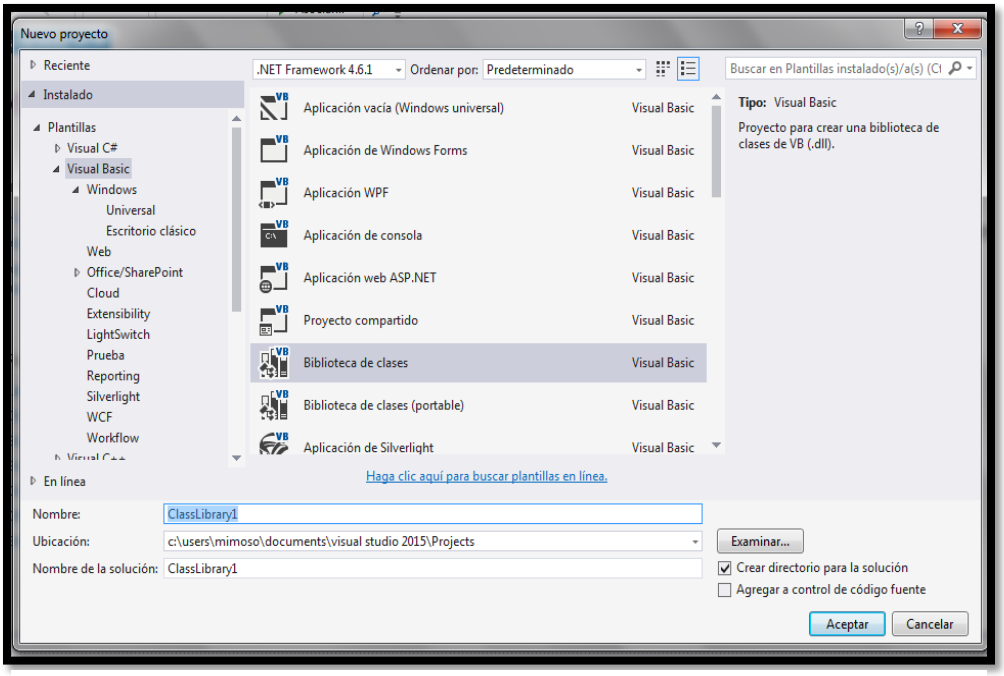

Figura 4. 7 Creación de proyecto de tipo librería de clases en Visual Basic.

Después de crear el proyecto del tipo librería de clases, hay que agregar una clase (Windows Forms) como se muestra en la figura 4.8.

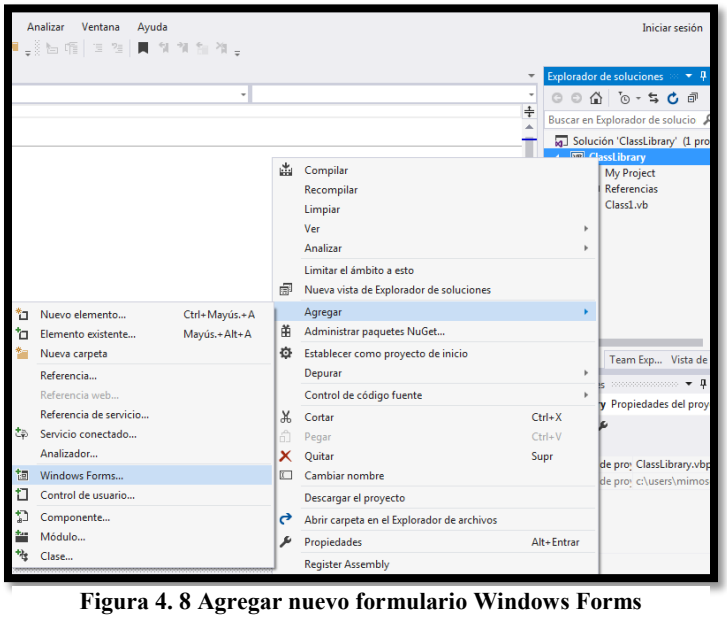

Figura 4. 8 Agregar nuevo formulario Windows Forms

En el diseñador del formulario se realiza el diseño de la ventana del objeto representante con los controles necesarios para mostrar la información más relevante del equipo cuando esté operando, tal como se muestra en la figura 4.9.

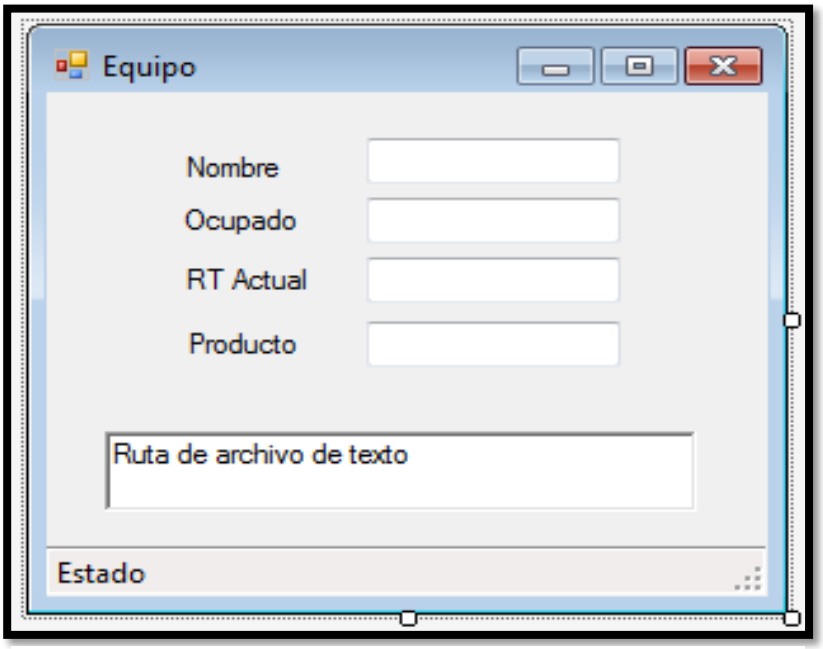

Figura 4. 9 Ventana de objeto representante.

Cuando el sistema de control crea los objetos representantes por reflexión envía como parámetros algunos datos obtenidos del esquema iMRP donde se incluyen el nombre, el estado, el recurso tiempo, la pieza que se va a procesar y el archivo de texto que se va a leer. Tal como se muestra en el listado 4.4.

Listado 4. 4 Parámetros enviados a la ventana objeto representante

```
Public Sub Datos(ByVal Nombre As String, Ocupado As Boolean, RT
As String,  producto As String)
        Me.Text = Nombre
        TextBox1.Text = Nombre
        TextBox2.Text = Ocupado
        TextBox3.Text = RT
        TextBox4.Text = producto
End Sub
```

Otra parte importante del desarrollo del objeto representante es habilitar la comunicación utilizando el servicio de cola de mensajes de Windows (MSMQ) como medio de comunicación, para ello es necesario agregar la referencia System.Messaging para poder utilizar los recursos de las colas de mensajes.

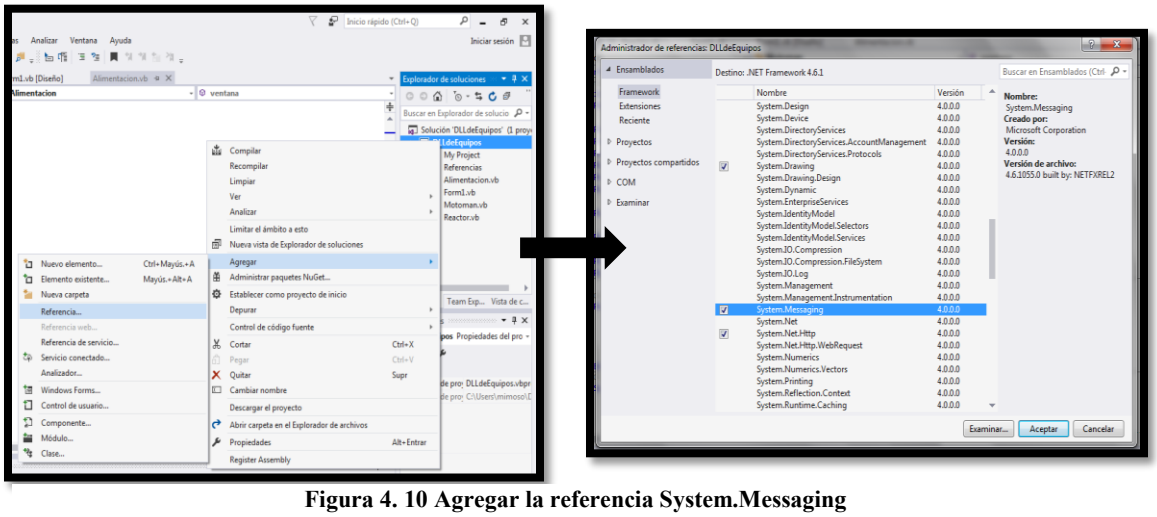

Figura 4. 10 Agregar la referencia System.Messaging

En la clase del objeto representante es necesario declarar una variable con la propiedad WithEvents de tipo MessageQueue para poder manejar el evento de recepción datos.

```
Public WithEvents obColaRecibir As MessageQueue
```

Para habilitar la recepción de mensajes se debe crear un nuevo objeto la variable obColaRecibir y a dicho objeto indicarle la dirección de la cola por la cual se recibirán los mensajes y después se debe iniciar el servicio para comenzar a leer los mensajes que lleguen a la cola con la instrucción obColaRecibir.BeginReceive(). Como se muestra en el listado 4.5

Listado 4. 5 Configuración para habilitar la recepción de mensajes por MSMQ.

```
Public Sub RecibirMSQM()
    obColaRecibir = New MessageQueue
    obColaRecibir.Path = RutaColaRecibir
    obColaRecibir.BeginReceive()
End Sub
```

Para procesar los mensajes recibidos se debe habilitar un evento en la variable asignada para recibir mensajes obColaRecibir, para ello se debe seleccionar el evento RecieveCompleted como se muestra en la figura 4.11

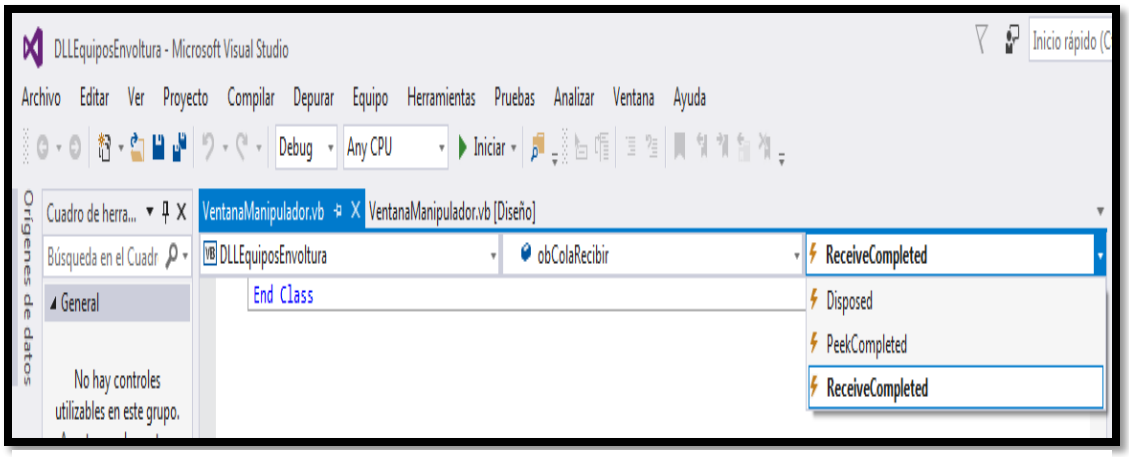

Figura 4. 11 Generar el evento ReceiveCompleted.

El evento RecieveCompleted genera un método el cual es invocado cada vez que se recibe un mensaje y dicho método es utilizado para procesar los mensajes provenientes de los objetos envoltura, dentro del método se crea y configura una variable de tipo Message en la cual se almacena el mensaje recibido para posteriormente ser procesado, tal como se muestra en el listado 4.6.

Listado 4. 6 Evento RecieveCompleted en el cual se procesan los mensajes recibidos.

```
Private Sub obColaRecibir_ReceiveCompleted(sender As Object, e As
ReceiveCompletedEventArgs) Handles obColaRecibir.ReceiveCompleted
 obColaRecibir.BeginReceive()
      Dim msgRecibido As New Message
      msgRecibido.Formatter = New XmlMessageFormatter
      CType(obColaRecibir.Formatter, XmlMessageFormatter).TargetTypeNames = New String()
{"System.String"}
      msgRecibido = obColaRecibir.EndReceive(e.AsyncResult)
      etiqueta = msgRecibido.Label
      cuerpo = msgRecibido.Body
 End Sub
```

La instrucción msgRecibido.Formatter = New XmlMessageFormatter sirve para serializar el cuerpo del mensaje, la instrucción obColaRecibir.EndReceive(e.AsyncResult) se utiliza para leer y eliminar el mensaje de la cola que fue recibido en el evento ReciveCompleted.

Cuando un mensaje es recibido, es necesario mostrar la información del mensaje en la ventana del objeto representante; para asignar a un control de la ventana un texto desde otro hilo diferente al usado por la ventana, es necesario asignar la propiedad CheckForIllegalCrossThreadCalls = False en el constructor del formulario.

Por otra parte, para enviar mensajes hacia los objetos envoltura ubicados generalmente en otra computadora, hay que crear una variable de tipo MessageQueue y asignarle como parámetro la ruta de la cola a la que le enviara el mensaje.

```
obColaEnviar = New MessageQueue(PathColaResponder)
obColaEnviar.Send(state, NombreEquipo)
```

4.1.4 Módulo 3: Aplicación Envoltura.

El desarrollo de la aplicación envoltura consiste en una aplicación de Windows Forms capaz de integrar objetos envoltura de manera dinámica en tiempo de ejecución utilizando la técnica de reflexión que ofrece el Framework de .NET y de establecer comunicación con una base de datos SQL Server para acceder a ella y registrar los tiempos en que se ejecuta cada operación. Esta aplicación es la encargada de instanciar objetos envoltura utilizando la técnica de reflexión por lo cual es necesario agregar la referencia System.Reflection como se muestra en la imagen 4.12, y así poder crear instancias del objeto envoltura de manera dinámica.

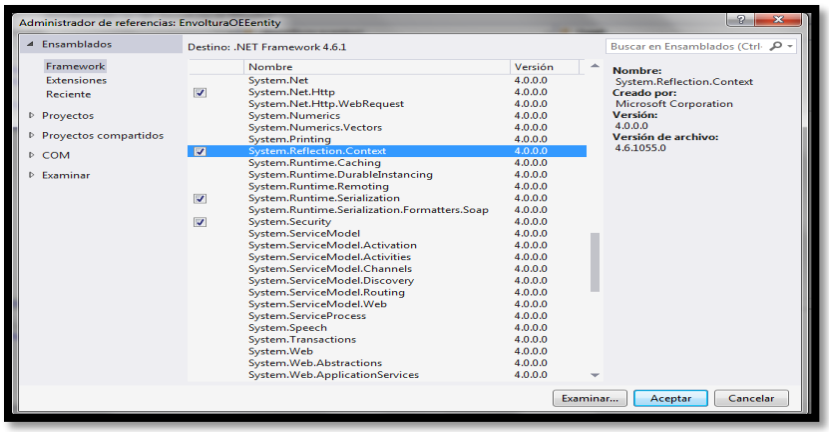

Figura 4. 12 Agregar Referencia System.Reflection al proyecto envoltura.

Para incorporar los objetos envoltura a la aplicación envoltura, primeramente es necesario crear las clases genéricas grafo y nodo tal y como se hizo en el módulo de sistema de control, estas clases son indispensables para poder deserializar un archivo XML en el cual está contenida la información de cada equipo.

Para crear los objetos envoltura se utiliza la herramienta de modelado iMRP donde sólo se crean nodos de tipo equipo tal como se muestra en la imagen 4.13.

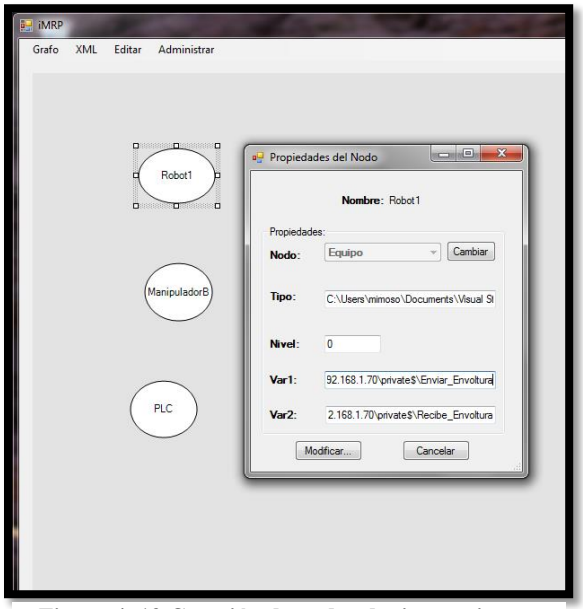

Figura 4. 13 Creación de nodos de tipo equipo en herramienta iMRP

En el listado 4.7 se muestra como crear una instancia de la clase envoltura por reflexión y obtener los datos del archivo XML los cuales se reciben como parámetros para crear el objeto envoltura.

Listado 4. 7 Creación de objeto envoltura por reflexión

```
Public Sub CrearEquiposReflexion(ByVal direccionDLL As String, idEquipo As String,
colaResponder As String, colaRecibir As String)
    Dim TestArray() As String = Split(direccionDLL, "\\")
        dll = TestArray(0)
        clase = TestArray(1)
        objAsm = Assembly.LoadFile(dll)
        objEqReflexion = objAsm.CreateInstance(clase)
        objEqReflexion.idEqu = idEquipo
        objEqReflexion.ColaResponder = colaResponder
        objEqReflexion.colaRecibir = colaRecibir
        objEqReflexion.propietario = Me
        objEqReflexion.MdiParent = Me
        listaEquipos.Add(objEqReflexion)
End Sub
```

4.1.4.1 *Incorporación Dinámica de Objetos Envoltura.*

El objeto envoltura se encarga de recibir las ordenes de operación provenientes del objeto representante a través del protocolo de comunicación de cola de mensajes de Windows, el objeto representante envía la ruta donde se encuentra un archivo de texto y el objeto envoltura lee las instrucciones contenidas en el archivo y con base en las instrucciones envía las ordenes al objeto propulsor el cual tiene comunicación directa con el controlador de equipo, además, cada vez que recibe una orden de operación registra en una base de datos la fecha y hora en que se recibió dicha orden de operación. Por otra parte, también recibe comandos del objeto propulsor cuando un equipo termina de realizar una operación; y registra en la base de datos la hora y fecha del final de operación y envía al objeto representante un mensaje para indicarle que un equipo termino de operar.

4.1.4.2 *Conexión con Base de Datos SQL Server.*

El objeto envoltura se conecta a una base de datos la cual es utilizada para almacenar los tiempos en que inician y terminan de operar los equipos, registrar tiempos muertos, etc.

Para acceder a la información de la base de datos se utilizó la tecnología de Entity Framework de .NET (Microsoft, Entity Framework, 2011) la cual permite manejar las tablas de la base de datos como objetos. Para realizar la conexión de la aplicación envoltura con la base de datos es necesario acceder al explorador de soluciones en la plataforma de visual Basic y dar clic secundario en el proyecto y elegir la opción agregar – Nuevo elemento y en la ventana que aparece seleccionar la opción Datos y el elemento ADO.NET Entity Data Model.

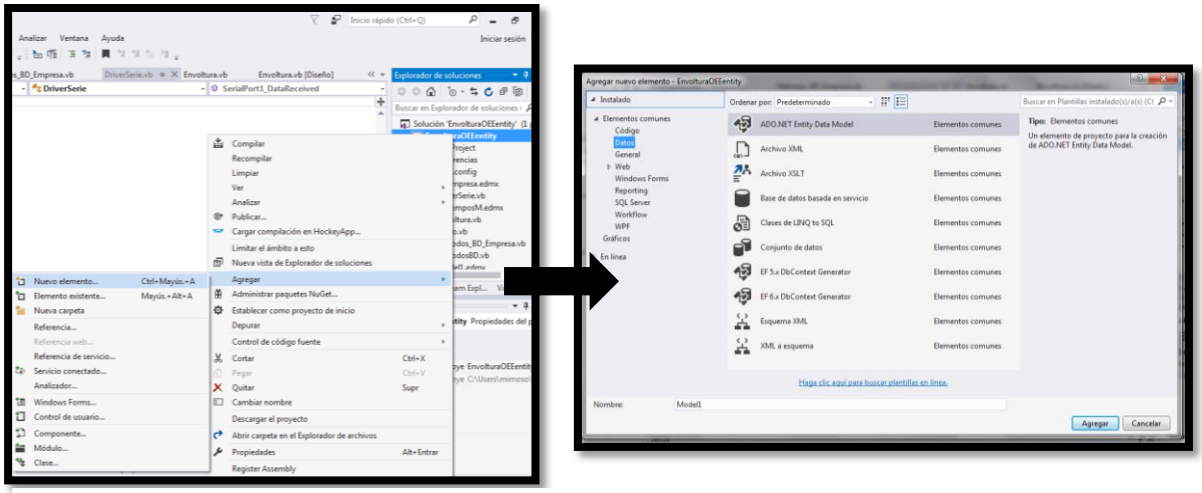

Figura 4. 14 Agregar Modelo de datos de Entity Framework.

En la figura 4.15 se muestran las ventanas que aparecen automáticamente, elegir la opción EF Designer desde base de Datos, seleccionar nueva conexión y elegir el nombre del servidor, el tipo de autenticación y el nombre de la base de datos.

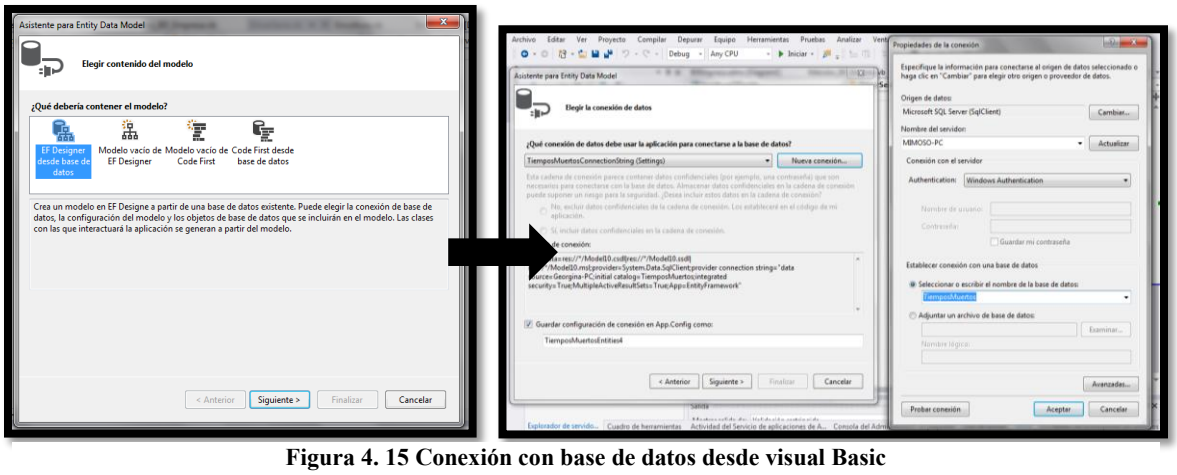

Figura 4. 15 Conexión con base de datos desde visual Basic

Finalmente, se deben seleccionar las tablas y procedimientos almacenados necesarios y se completara la conexión con la base de datos desde Visual Basic.

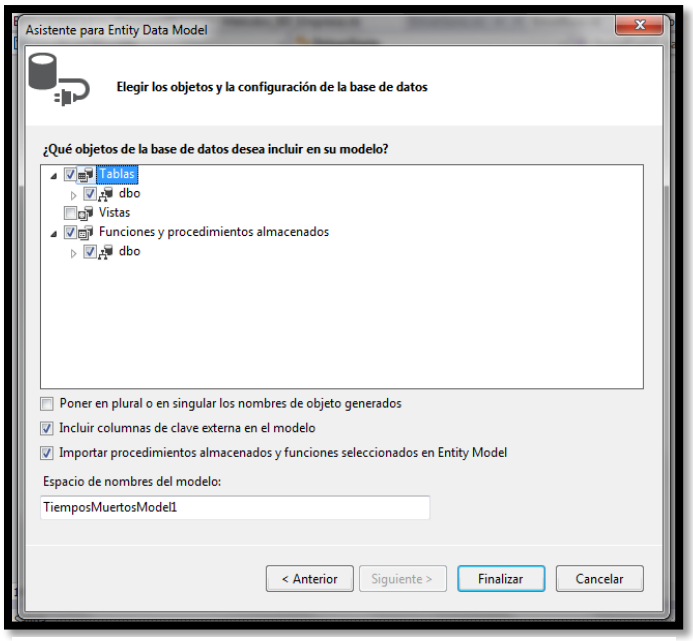

Figura 4. 16 Selección de tablas y Procedimientos Almacenados.

Al terminar el proceso se obtiene en el proyecto un nuevo elemento con extensión .edmx, en el cual se encuentra el diagrama entidad relación y las clases de cada una de las tablas.

En el objeto envoltura se crea una clase en la cual se realiza la conexión con la base de datos invocando a la clase Entities lo cual vuelve posible el acceso a la misma, listado 4.8, luego, para poder realizar consultas se utiliza Linq (Language Integrated Query), que son un conjunto de herramientas del visual Studio que permite realizar consultas a distintas fuentes de datos tales como bases de datos, objetos, XML, entre otros. (Microsoft, 2017).

Listado 4. 8 Conexión a la base de datos y consulta utilizando Linq.

```
Private dbTM As New EmpresaEntities()
Public Function ObtenerNombreEquipoBD(ByVal idEqu As String)
        Dim NombreEqu = (From E In dbTM.Equipos
                         Where E.id_Equipo = idEqu
                         Select E.nombre).Single
        Return NombreEqu
End Function
```

4.1.5 Módulo 4: Objeto Envoltura.

En esta sección se presentan el desarrollo del objeto envoltura el cual consiste en crear un proyecto de tipo de librería de clases y dentro del proyecto crear una clase de tipo Windows Forms en la cual se incluirán los comandos para establecer la comunicación a través de cola de mansajes, las propiedades y métodos para insertar registros en la base de datos y las instrucciones para leer el archivo de texto.

Para establecer la comunicación por el protocolo de cola de mansajes es necesario realizar las mismas configuraciones realizadas en el objeto representante, se debe agregar la referencia System.Messaging para poder enviar y recibir mensajes, también es necesario declarar algunas variables con la propiedad WithEvents de tipo MessageQueue las cuales servirán para manejar los eventos de recepción y envió de datos.

Luego se debe crear un nuevo objeto de la variable asignada para recibir los mensajes e iniciar el servicio para comenzar a leer los mensajes que lleguen a la cola con la instrucción BeginReceive().

```
obColaRecibir.BeginReceive().
```

Para procesar los mensajes recibidos es necesario crear el evento recieveCompleted en la variable obColaRecibir. Lo cual genera automáticamente el método obColaRecibir_ReceiveCompleted(sender As Object, e As ReceiveCompletedEventArgs) Handles obColaRecibir.ReceiveCompleted dentro del cual se procesan los mensajes recibidos, los cuales están divididos en dos partes, la etiqueta la cual contiene información del objeto propulsor al cual se manda operar; y el cuerpo en donde está contenida la ruta del archivo de texto que se va a leer.

En el listado 4.9 se muestran los métodos a los que se accede cuando el objeto propulsor recibe un mensaje en los cuales se realiza la lectura del archivo de texto y la inserción de registros a la base de datos.

Listado 4. 9 Método invocado cuando el objeto propulsor recibe un mensaje

```
Public Sub Operando(ByVal RutaArchivo As String)
    LeerArchivo(RutaArchivo)
    MetodosDeBD.InsertarInicio(IdSubLote, IdFase, Fase)
End Sub
```

4.1.5.1 Archivo de Texto.

El objeto propulsor es el encargado de leer el archivo de texto en el cual están contenidas las instrucciones de operación del equipo físico, sin embargo, para poder leer el archivo de texto es necesario importar el espacio de nombres System.IO, luego se debe crear una variable de tipo StreamReader y pasarle como parámetro la ruta del archivo, también hay que crear una variable de tipo String para guardar los datos que se leen en el archivo línea por línea. Finalmente se crea un ciclo para recorrer todas las líneas del archivo y dentro del ciclo se procesa la información del archivo, tal como se muestra en el listado 4.10.

Listado 4. 10 Método para leer Archivo de texto

```
Public Sub LeerArchivo(ByVal ruta As String)
    Dim Texto() As String
    Dim objreader As New StreamReader(ruta)
    Dim sLine As String = objreader.ReadLine()
    While sLine Is Nothing = False
            If sLine.Contains("Origen") Then
                Texto = Split(sLine, ":")
                Dim Origen = Texto(1)
                TextBox1.Text = Origen
            End If
        sLine = objreader.ReadLine()
    End While
End Sub
```

4.1.5.2 Base de Datos.

En la aplicación envoltura se encuentra la conexión con la base de datos y ahí mismo se encuentra una clase la cual fue desarrollada para realizar las consultas necesarias para insertar y consultar la información de la base de datos. Cuando la aplicación envoltura crea una instancia del objeto envoltura le asigna a una de las variables del objeto envoltura una instancia de la clase donde se encuentran los métodos para acceder a la base de datos, de esta manera el objeto propulsor puede consultar, insertar y modificar la información de la base de datos según sea requerido.

4.1.5.3 Incorporación Dinámica de Objeto Propulsor

Para incorporar el objeto propulsor (driver) dinámicamente por reflexión al objeto envoltura es necesario crear la instancia de la clase propulsor o driver utilizando la propiedad Assembly al igual que se hizo anteriormente con el objeto envoltura, tal como se muestra en el listado 4.11.

Listado 4. 11 Creación de objeto propulsor por reflexión

```
Dim TestArray() As String = Split(direccionDLL, "\\")
If TestArray.Length > 1 Then
    dll = TestArray(0)
    clase = TestArray(1)
    objAsm = Assembly.LoadFile(dll)
    objDriverReflexion = objAsm.CreateInstance(clase)
End If
```

4.1.6 Módulo 5: Objeto Propulsor.

El desarrollo del objeto propulsor (driver) al igual que el objeto envoltura y el objeto representante se hace creando un proyecto de tipo librería de clases, donde se crean clases que contienen métodos e instrucciones para configurar la comunicación con el equipo físico y una vez que el proyecto está listo es necesario desarrollar una clase que permita la comunicación con el dispositivo físico.

El objeto driver contiene métodos para establecer la conexión con el dispositivo físico, además para enviar ordenes de operación las cuales se encuentran localizadas en un archivo de texto y para recibir señales que indican que el equipo termino de operar.

4.1.7 *Comunicación Objeto representante - Objeto envoltura*

Al ejecutar el sistema de control y el objeto envoltura se crean las instancias de objetos representantes y los objetos propulsores respectivamente, durante la instanciación cada objeto recibe como parámetros la dirección de la cola de mensaje por la cual enviará y recibirá datos de la envoltura.

Para poder establecer la comunicación entre el objeto representante y el objeto envoltura fue necesario crear una cola de mensajes para cada objeto, para esto es necesario habilitar el servicio de colas de mensajes siguiendo la ruta panel de control/programas/programas y características/Activar o desactivar las características de Windows y ahí seleccionar Microsoft Message Queue (MSMQ) Server, tal como se muestra en la figura 4.17.

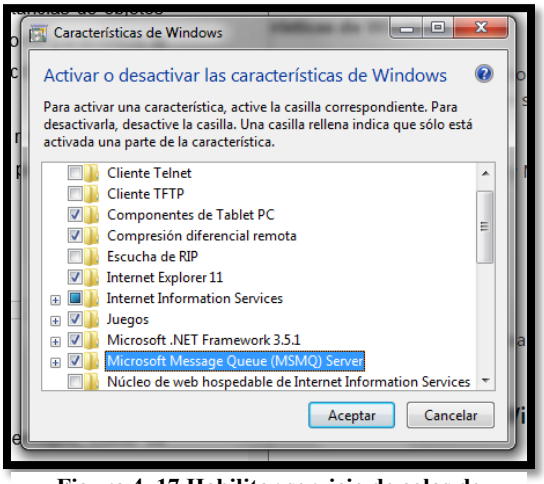

Figura 4. 17 Habilitar servicio de colas de mensajes de Windows.

Luego es necesario ir al explorador de servidores de visual Studio y crear una cola para el objeto representante y otra para cada objeto propulsor. Para crear una cola privada hay que introducir el siguiente formato de ruta .\Private$\nombreDeCola en caso de que los objetos representante y propulsor se encuentren en la misma PC, o el siguiente formato en caso de que se encuentren en diferente PC FormatName:Direct=TCP:DireccionIP\private$\NombreDeCola como se muestra en la figura 4.18.

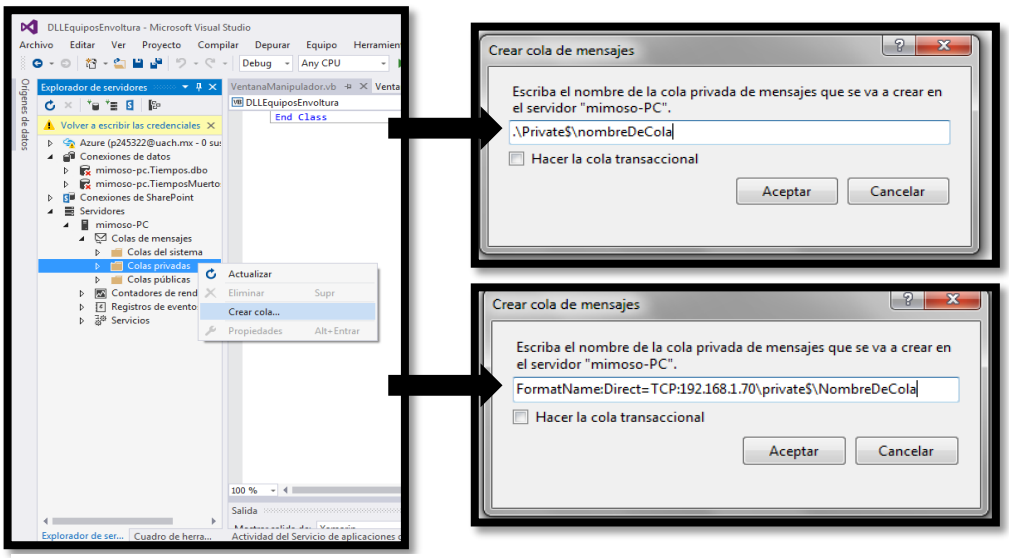

Figura 4. 18 Crear Colas privadas desde visual Basic.

4.2 APLICACIÓN DEL CALCULO OEE.

Uno de los puntos de la solución consiste en manejar funciones administrativas, es decir, las empresas buscan medir y coordinar de manera óptima los procesos, en este sentido se desarrolló una aplicación para manejar el parámetro de desempeño denominado OEE el cual permite calcular el rendimiento de los equipos en un piso de producción.

En la figura 4.19 se presenta un diagrama de bloques para crear una aplicación capaz de calcular el OEE y de generar reportes.

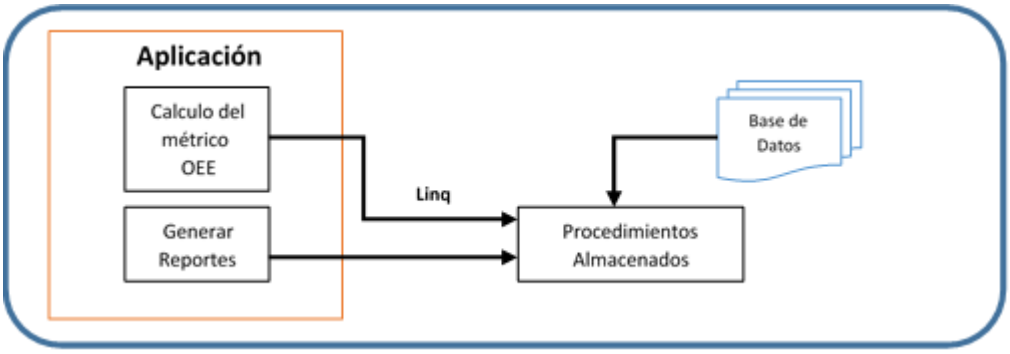

Figura 4. 19 Diagrama a bloques de Aplicación de reportes OEE

La aplicación está dividida en dos módulos, el primer módulo es el cálculo del OEE y el segundo es la creación de reportes, ambos módulos consultan la base de datos, la cual es la misma que se implementó en el sistema informático de control, para poder calcular el OEE y para generar reportes con la información contenida en ella. Dentro de la base de datos se programaron varios procedimientos almacenados los cuales son invocados por ambos módulos de la aplicación utilizando consultas Linq.

La aplicación se conecta a la misma base de datos que se conecta el objeto envoltura utilizando Entity Framework, para realizar la conexión se deben de seguir los mismos pasos que se realizaron en la envoltura.

4.2.1 Cálculo del OEE.

Como se mencionó en el capítulo anterior, las empresas de manufactura buscan medir el tiempo real que trabajan los equipos y detectar las causas que provocan que no operen óptimamente y así poder mejorar el rendimiento de los mismos, para ello utilizan el métrico de desempeño OEE que es una fórmula simple en la cual se calcula la disponibilidad, la eficiencia y la calidad con la que los equipos realizan los procesos.

Para realizar el cálculo del OEE la aplicación toma los datos necesarios desde la base de datos tales como la duración del tiempo operado de cada equipo, los tiempos muertos causados por paros planificados y no planificados, entre otros. Para acceder a la información de la base de datos se realizaron consultas Linq (Language integrated query) ya que en una consulta Linq siempre se trabaja con objetos lo cual simplifica el acceso a los datos.

En el listado 4.12 se muestra un ejemplo de una consulta a la base de datos utilizando Linq.

Listado 4. 12 Consulta a la base de datos para obtener los equipos

```
Public Function ObtenerNombreEquipoBD(ByVal
idEqu As String)
    Dim NombreEqu = (From E In dbTM.Equipos
                Where E.id_Equipo = idEqu
                Select E.nombre).Single
    Return NombreEqu
End Function
```

Al iniciar la aplicación se realiza una consulta a la base de datos para obtener todos los equipos disponibles los cuales son mostrados en un combobox, luego, para realizar el cálculo del OEE se debe seleccionar un equipo y un rango de fechas (día, semana, mes) ya que al realizar las consultas se obtienen los datos generados entre esas fechas.

4.2.2 Reportes.

En esta sección se muestra el desarrollo de reportes los cuales son informes gráficos que sirven como herramienta de apoyo para analizar el comportamiento de los equipos y los procesos en un piso de producción y para la toma de decisiones necesarias para el incremento del OEE.

Para elaborar los reportes se utilizó una herramienta de Visual Basic .Net llamada ReportViewer la cual permite elaborar informes de manera interactiva con una gran calidad de presentación, dicha herramienta se encuentra en el cuadro de herramientas en la sección de generación de informes.

Primeramente en el formulario de la aplicación se debe arrastrar el control ReportViewer, tal como se muestra en la imagen 4.20

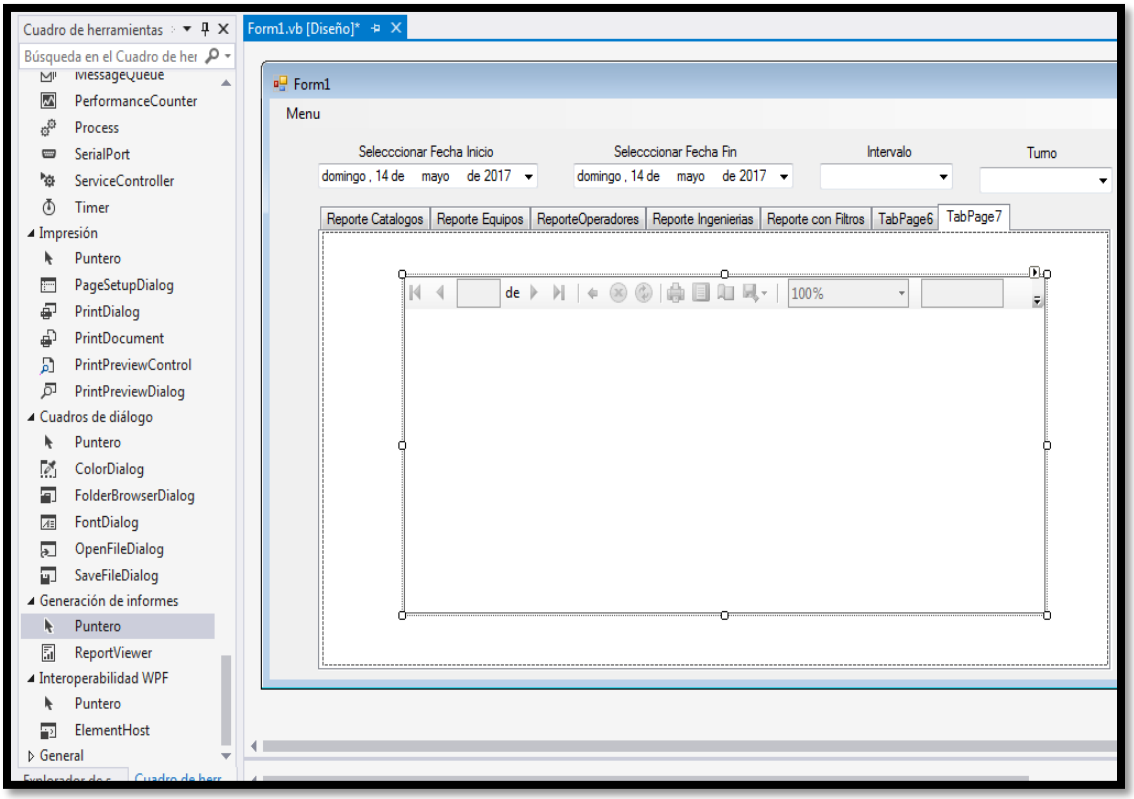

Figura 4. 20 Control Report Viewer.

Luego es necesario agregar un nuevo informe, para ello hay que ir al menú proyecto y agregar un nuevo elemento e ir a la sección Reporting y seleccionar la opción Informe, tal como se muestra en la figura 4.21, lo cual genera un nuevo elemento con extensión .rdlc.

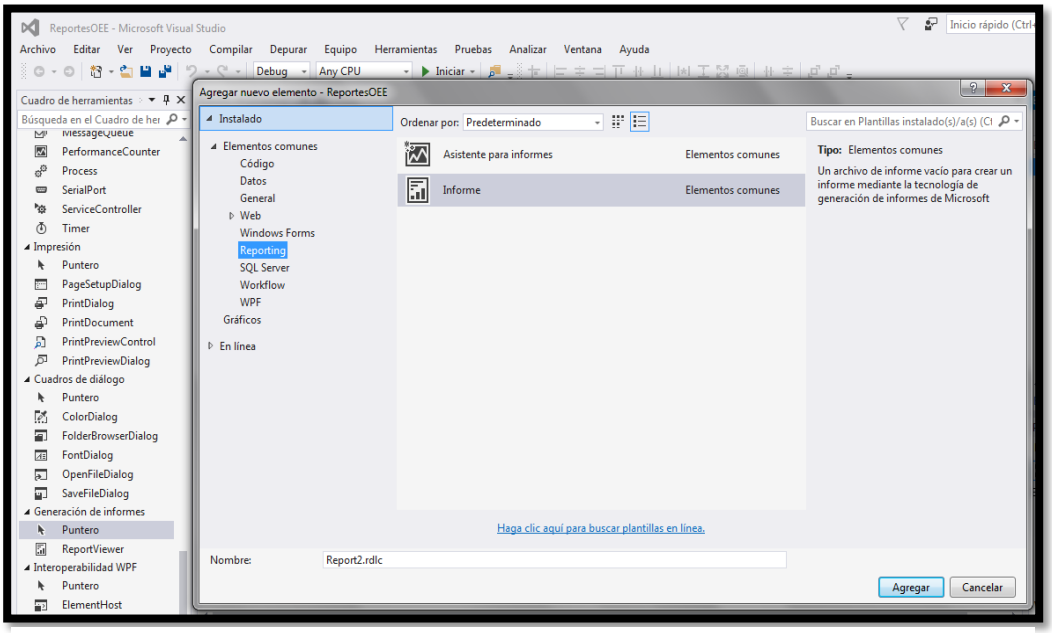

Figura 4. 21 Agregar nuevo Informe.

El siguiente paso es enlazar la base de datos con el reporte para poder acceder a los datos de la misma, para esto hay que seleccionar el nuevo elemento creado y en el menú datos del informe agregar un nuevo origen de datos y seleccionar la base de datos de la cual se obtendrá la información, figura 4.22.

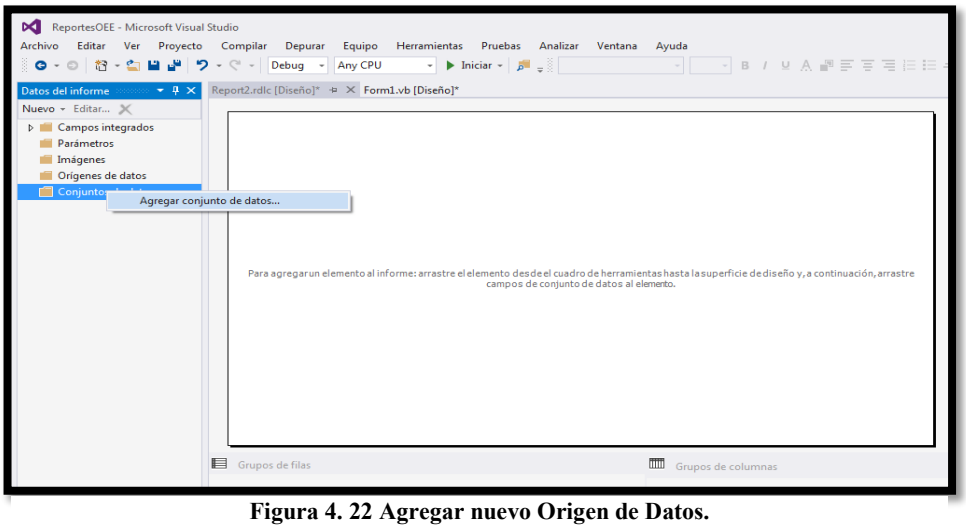

Figura 4. 22 Agregar nuevo Origen de Datos.

Ahora se deben de agregar las tablas o los procedimientos almacenados necesarios como conjuntos de datos y con base a ellos se podrán crear tablas, gráficas y otros elementos en el informe. Para agregar las tablas o gráficas es necesario acceder al cuadro de herramientas y arrastrar dichos controles al informe, figura 4.23, luego para relacionar el conjunto de datos con los controles hay seleccionar el conjunto de datos en la tabla o gráfica, tal como se muestra en la figura 4.24.

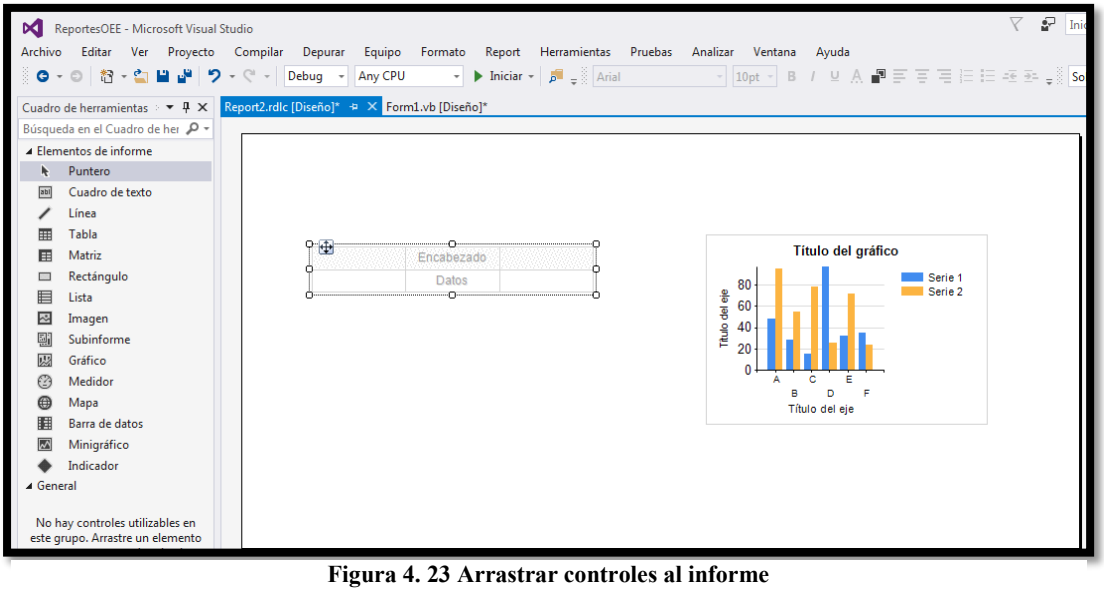

Figura 4. 23 Arrastrar controles al informe

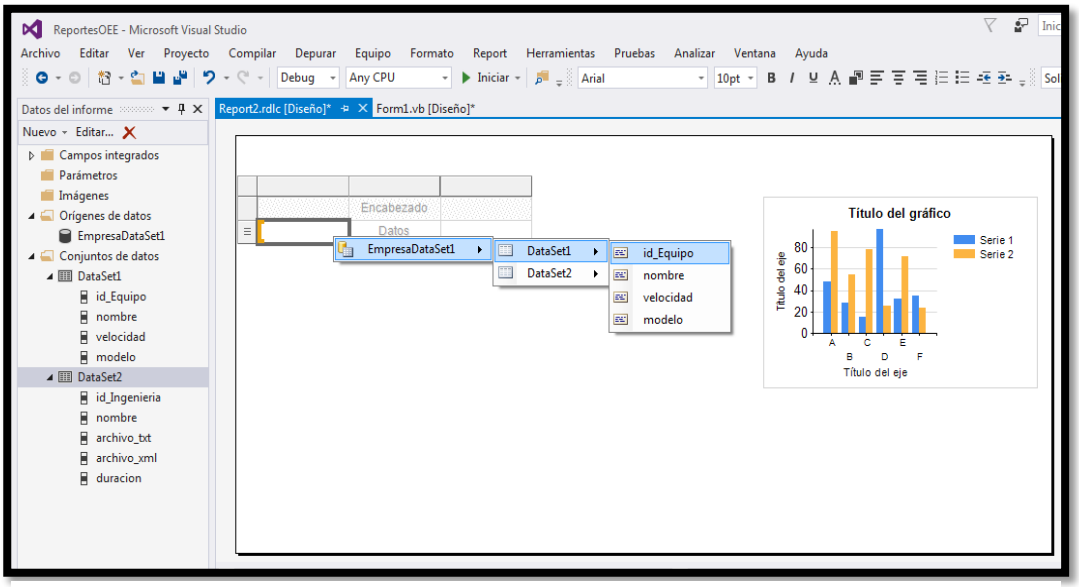

Figura 4. 24 Relacionar conjunto de datos con tablas y gráficas.

Para poder mostrar el informe en el formulario principal, se debe de seleccionar desde la herramienta ReportViewer, tal como se muestra en la figura 4.25.

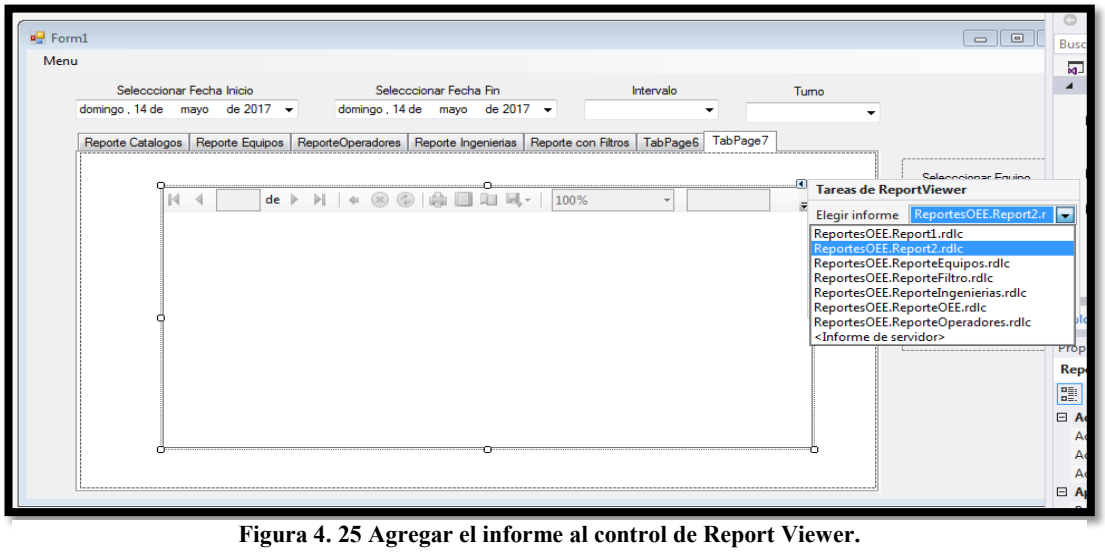

Figura 4. 25 Agregar el informe al control de Report Viewer.

Finalmente al agregar el informe a la herramienta de ReportViewer se genera código en el constructor del formulario el cual contiene la información de las tablas y procedimientos almacenados agregados en el informe, sin embargo, en ocasiones necesita ser editado con parámetros que deben de ser llenados manualmente.

```
Me.IngenieriasTableAdapter.Fill(Me.EmpresaDataSet1.Ingenierias)
Me.ReportViewer7.RefreshReport()
```

CAPÍTULO 5. VALIDACIÓN Y RESULTADOS DE LA SOLUCIÓN PROPUESTA

Con el propósito de probar el alcance del sistema informático de control lanzador de órdenes y la capacidad del sistema para incorporar funciones, (módulo para calcular el métrico de desempeño OEE), desarrollado en el capítulo anterior, se diseñó una célula de manufactura a nivel de laboratorio utilizando robots didácticos "Robobuilder", robot Motoman UP-20 y PLC; posteriormente, se realizó una estancia en una empresa real orientada a la fabricación de productos químicos, para implementar el sistema en su entorno de producción, en paralelo al sistema de control real de producción de la misma. Es importante destacar que en el primer ejercicio de validación se trata de un sistema de fabricación discreta y en el segundo caso es un sistema de fabricación por lotes. A continuación se describe cada uno de los ejercicios de validación mencionados.

5.1 VALIDACIÓN DEL SISTEMA LANZADOR A NIVEL LABORATORIO

Como parte de la validación del sistema lanzador de órdenes a nivel laboratorio se desarrolló una célula de manufactura con distintos equipos, con el propósito de simular el lanzamiento de órdenes y probar la flexibilidad del sistema en la integración de nuevos equipos.

La célula de manufactura se formó utilizando equipos existentes en el laboratorio de automática de la facultad de ingeniería de la UACH, tales como, robots didácticos Robobuilder, Robot industrial Motoman UP-20 y un PLC Allen-Bradlley, en la figura 5.1 se presenta un diagrama de los equipos incluidos en la célula de trabajo.

Los equipos de producción (en este caso los robots Motoman, Robobuilder y el PLC), se integran al control utilizando el mecanismo de acoplamiento débil (Acosta, 2016), que consiste en un objeto representante (operando en el mismo espacio de direcciones que el sistema de control -mismo programa -), un objeto envoltura y un objeto propulsor.

La operación del sistema de control requiere información sobre las particularidades del sistema físico de producción (equipos, piezas del producto, procesos a realizar a cada pieza en cada equipo, secuencia de operación de los equipos, programa a ejecutar en cada equipo, entre otras particularidades). Las particularidades son abstraídas (modeladas) en con base en la técnica gráfica iMRP. El modelo iMRP del sistema es almacenado en un archivo XML para su interpretación por el sistema de control. Con base en el modelo, el sistema de control crea un objeto representante por cada equipo del piso de producción (robots y PLC). Las órdenes de operación dirigidas a los equipos de producción son lanzadas al objeto

representante respectivo, así el sistema de control se dirige a un objeto (representante) que se encuentra en la misma aplicación del sistema de control, de manera tal que las características del equipo real de producción son transparentes al sistema de control. Posteriormente, el objeto representante es el encargado de transmitir la orden de operación hacía el equipo de producción.

El proceso de validación del sistema consiste en enviar órdenes de operación con base en el modelo de particularidades iMRP desde los objetos representantes creados por el sistema de control (lanzador de órdenes) a los distintos objetos envolturas, utilizando el protocolo de comunicación de servicio de cola de mensajes de Windows (MSMQ), comunicación que generalmente es de naturaleza remota.

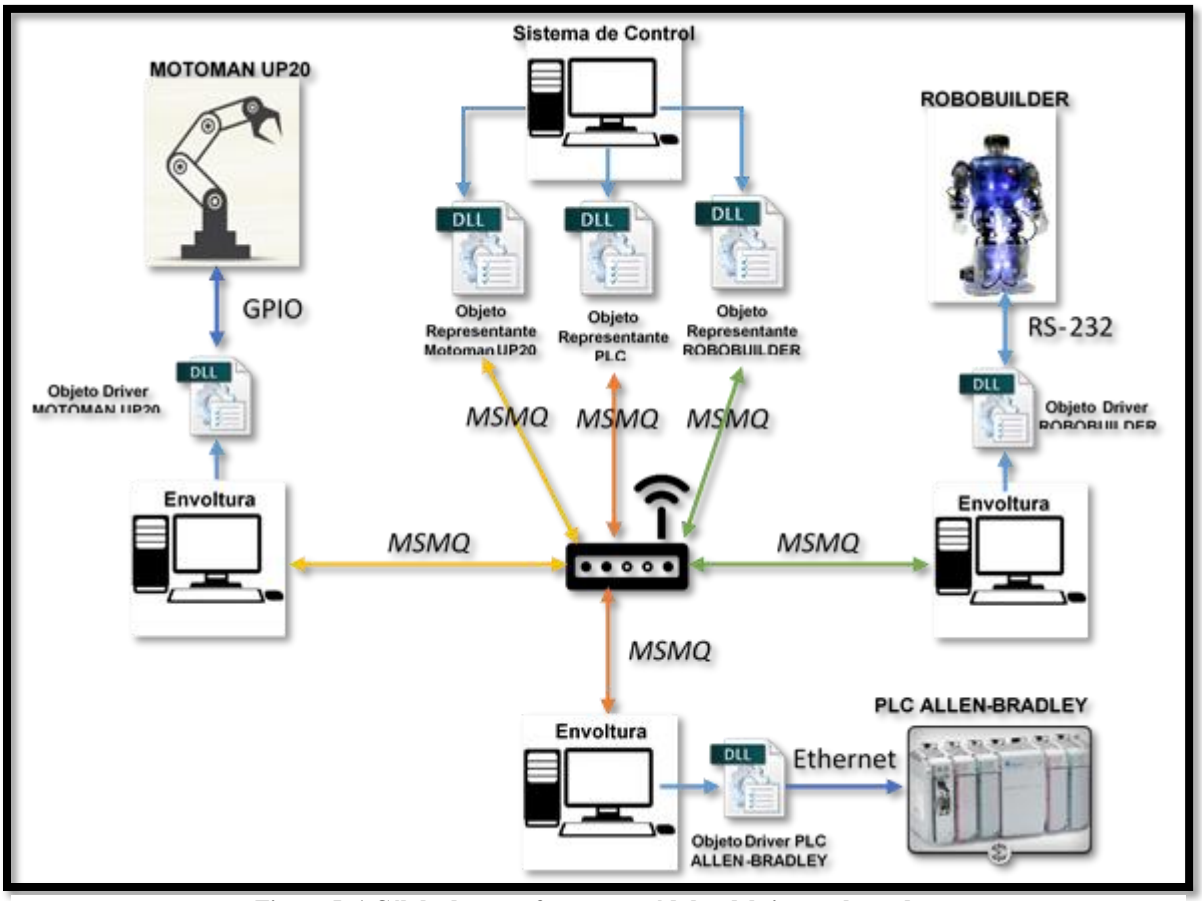

Figura 5. 1 Célula de manufactura y módulos del sistema lanzador.

El ejercicio de validación se basa en la integración del sistema lanzador de órdenes con los equipos de una célula de manufactura formada por un robot Motoman UP-20, un PLC Allen-Bradlley Controllogix y Robots didácticos Robobuilder, figuras 5.2, 5.3 y 5.4. Las operaciones de la célula consisten en utilizar el robot industrial para manipular y apilar tablillas en una estación de pruebas, es decir, realiza acciones de colocar y retirar las tablillas, por otra parte se emplea el PLC para simular la ejecución de varios programas y los robot didácticos se utilizan para simular el procesamiento de piezas.

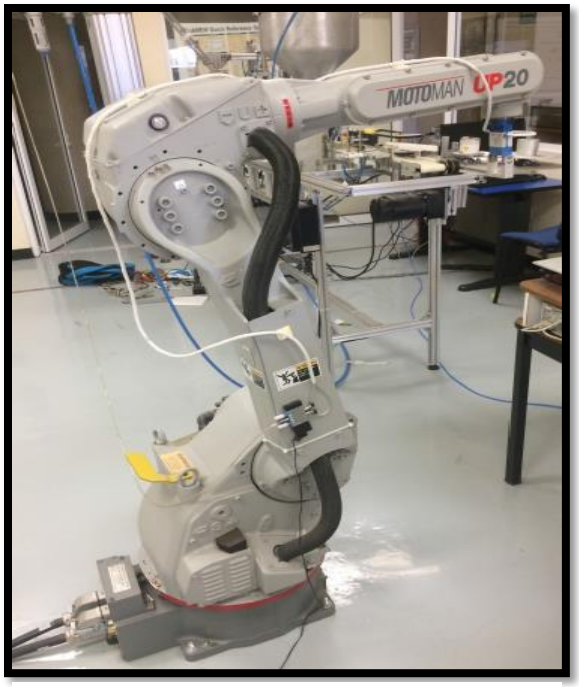

Figura 5. 3 Robot Industrial Motoman UP-20.

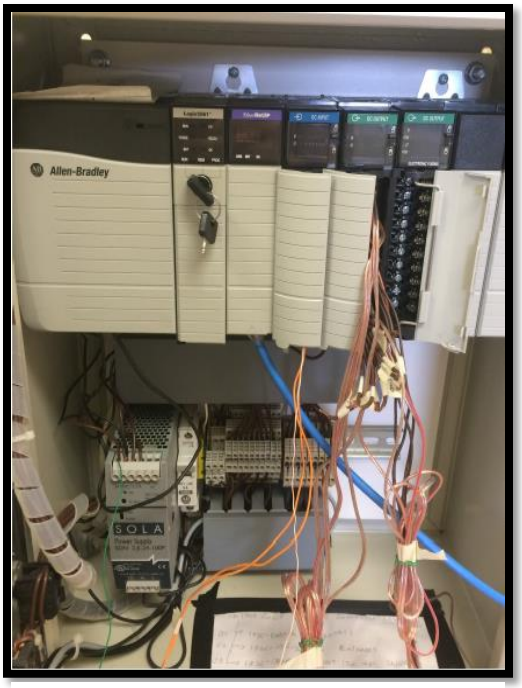

Figura 5. 2 PLC Allen-Bradlley Control logix.

Figura 5. 4 Robots didácticos Robobuilder.

La programación de los equipos y la creación de los archivos de texto con las instrucciones específicas para cada equipo se realizaron previamente a la operación del sistema, primero se crearon varias rutinas en el robot industrial Motoman las cuales son ejecutadas dependiendo de las entradas de propósito general que se encuentren activas, después, se realizó un programa en el PLC Control Logix 5000, en el cual se crearon ciertas etiquetas OPC para controlar las entradas y salidas, finalmente, para controlar los robots didácticos Robobuilder se programaron distintas rutinas (utilizando el software "Robobuilder Tool"), las cuales son activadas por medio del puerto serial RS-232.

Para establecer la comunicación desde el sistema de control con cada uno de los equipos fue necesario crear un objeto envoltura y un objeto driver distinto por cada equipo debido a que cada uno maneja un protocolo de comunicación físico particular, tal como se muestra en la figura 5.5.

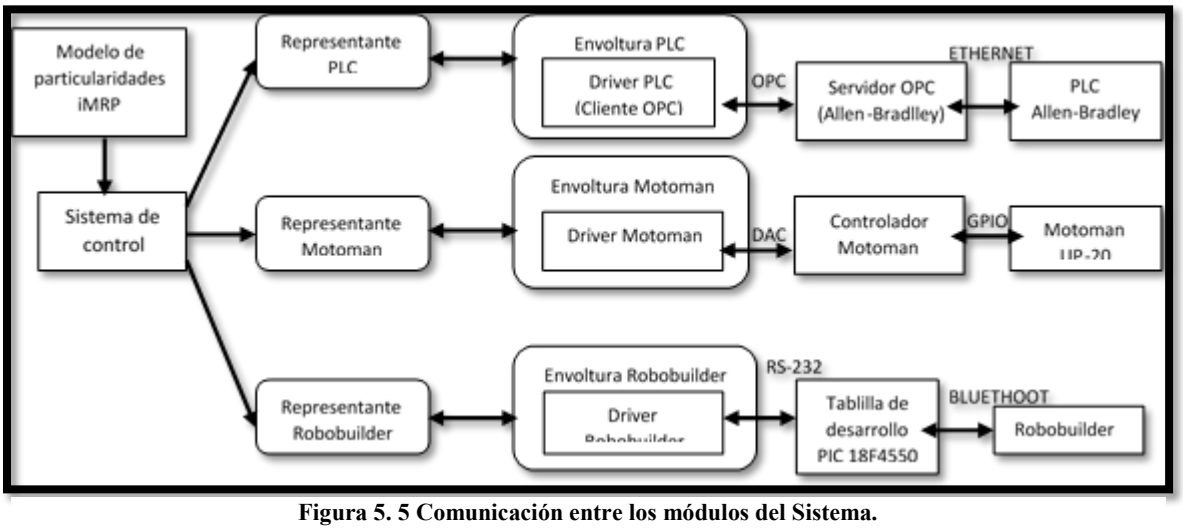

Figura 5. 5 Comunicación entre los módulos del Sistema.

La comunicación con el robot Motoman se hace a través de una tarjeta DAC de National Instruments la cual permite configurar y activar ciertas entradas de propósito general en el controlador del robot para que ejecute determinadas rutinas, figura 5.6, para ello es necesario agregar integrar librerías de National Instruments (DAQmx) al objeto propulsor. Para establecer la comunicación con el controlador del PLC de manera remota utilizando el protocolo Ethernet fue necesario crear un cliente y un servidor OPC y crear ciertas etiquetas OPC para controlar las entradas y salidas en el PLC, para ello fue necesario agregar la librería de OPC (OPCAutomation) en el objeto driver. Finalmente, para lograr

establecer la comunicación con los controladores de los robots Robobuilder fue necesario emplear microcontroladores PIC, como intermediarios entre el objeto driver y los controladores de los robot, tal como se muestra en la figura 5.7 y 5.8, la comunicación entre el objeto propulsor y los PIC se realizó utilizando el protocolo RS-232 y finalmente la comunicación entre el controlador PIC y el robot se logró utilizando bluethoot.

Figura 5. 6 Controlador Motoman UP-20.

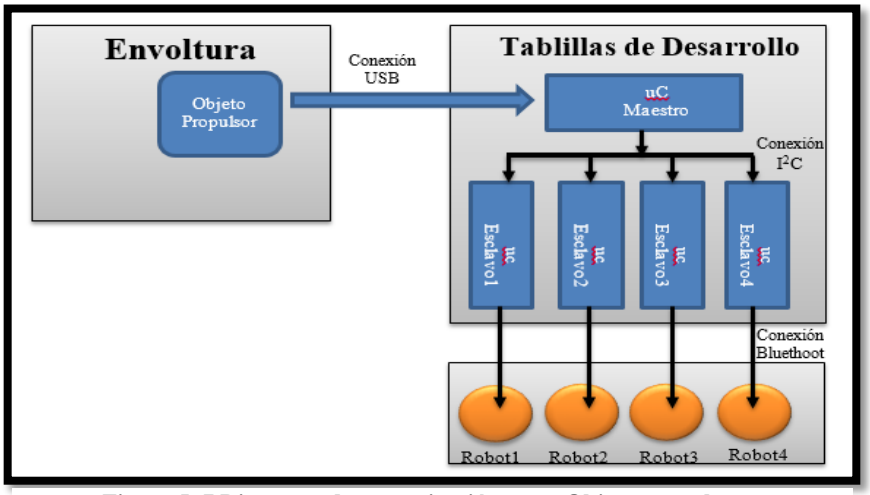

Figura 5. 7 Diagrama de comunicación entre Objeto propulsor y Microcontrolador PIC 18F4550.

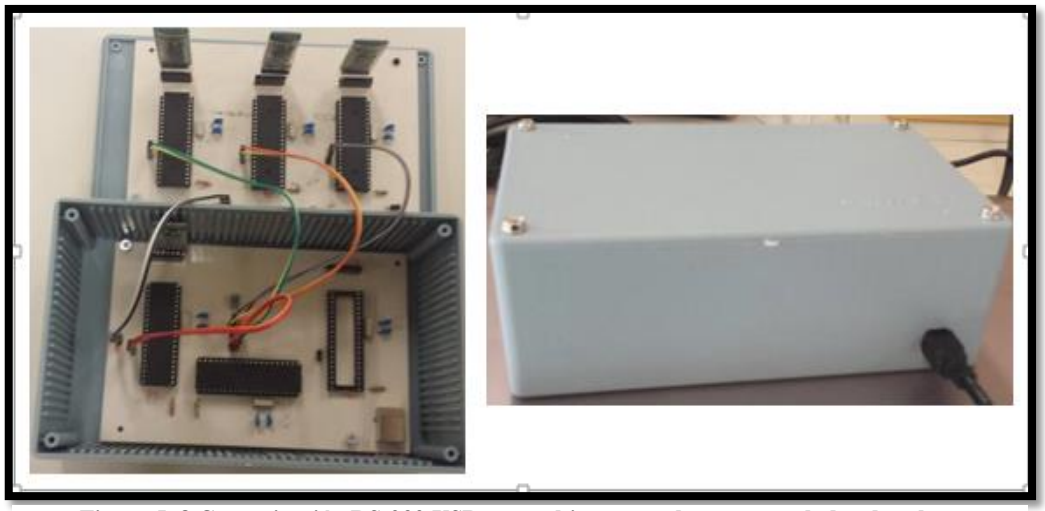

Figura 5. 8 Comunicación RS-232 USB entre objeto propulsor y controlador de robots Robobuilder

El proceso para el lanzamiento de órdenes a los equipos inicia con la abstracción de particularidades del proceso en iMRP. Posteriormente el modelo se representa en un archivo XML. El primer paso para el lanzamiento de órdenes consiste en ejecutar la aplicación de control, figura 5.9 y las aplicaciones envoltura que generalmente están ubicadas en diferentes PC, al ejecutar el sistema de control, se selecciona un archivo .XML donde se encuentra toda la información del modelo grafico iMRP, en dicho archivo están contenidos los equipos, operaciones, piezas y los recursos tiempo que son los que indican la secuencia de los pasos a seguir. Al de-serializar e interpretar el archivo, el sistema lanzador crea los objetos representantes y asigna el orden en que se enviaran las operaciones.

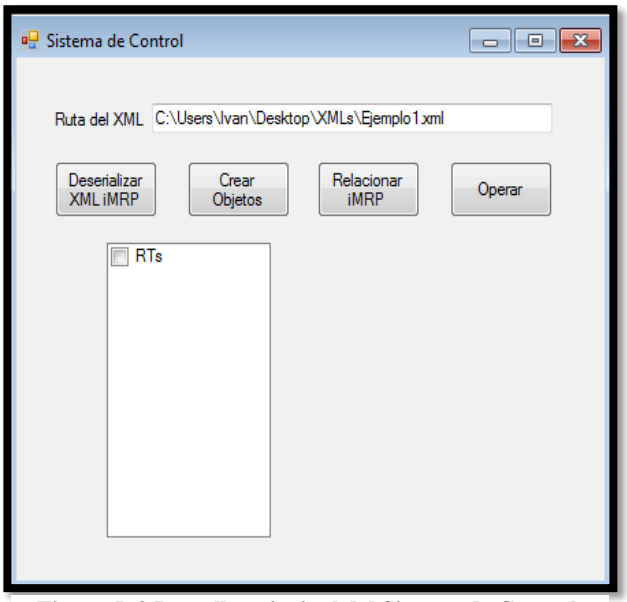

Figura 5. 9 Pantalla principal del Sistema de Control.

Por otra parte, al ejecutar cada objeto envoltura se genera la instancia del objeto propulsor (driver) el cual es el responsable de manejar los detalles en la comunicación con el controlador del equipo correspondiente.

Luego de iniciar las aplicaciones envoltura y el sistema de control, el sistema lanzador envía el comando operar a cada uno de los objetos representantes donde se incluye la ubicación y el nombre del archivo de texto en donde se describe el programa a realizar, luego el objeto representante re-envía a la envoltura correspondiente el mensaje recibido por el sistema lanzador a través del servicio de cola de mansajes de Windows (MSMQ). Cuando la envoltura recibe el mensaje, busca el archivo de texto en la dirección indicada en el mensaje y envía los comandos contenidos en el archivo al objeto driver el cual re-envía dichos comandos por el medio físico que acepta el controlador del equipo. Cuando el equipo físico termina de realizar una operación o ejecutar alguna rutina, envía una señal a través del medio físico al objeto driver y este re-envía al objeto envoltura el mensaje de fin de operación y la envoltura envía a través de MSMQ un mensaje al objeto representante indicándole el fin de operación del equipo y finalmente este le indica al sistema de control el fin de operación. Luego el sistema de control vuelve a enviar otra orden de operación a los equipos disponibles con base en el modelo.

5.2 SISTEMA LANZADOR DE ÓRDENES EN UN PISO REAL DE PRODUCCIÓN.

Como parte de la validación del trabajo el sistema se implementó a manera de prototipo en el piso de producción de una empresa de productos Químicos. Para la implementación se solicitaron a la empresa la información de los procesos (particularidades) para implementar el sistema de control lanzador de órdenes y la aplicación de reportes y cálculo del OEE en su piso de producción. Ambos sistemas se adaptaron a los procesos de producción.

La validación se realizó en una célula del piso de producción que está compuesta por distintos equipos tales como, reactores en los cuales se procesan los productos, bombas y tuberías utilizadas para manipular y transportar las materias primas, tanques, tolvas y depósitos que cumplen la función de almacenar el producto final, tal como se muestra en la figura 5.10

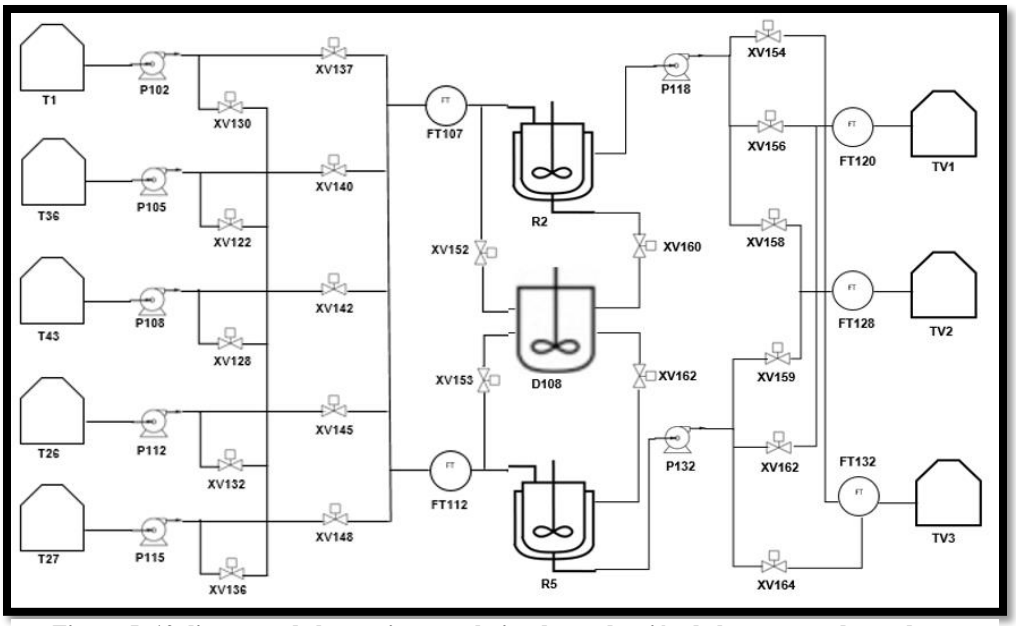

Figura 5. 10 diagrama de los equipos en el piso de producción de la empresa de producto químicos.

Debido a que los equipos que conforman la célula no son automáticos ni programables, su operación es dirigida por operadores, los cuales se encargan de seguir paso a paso las instrucciones o fases de una ingeniería (receta) que está en una hoja impresa para realizar el proceso de producción en los equipos de manera manual. Por tal motivo, se optó por abstraer la ingeniería o receta en un modelo grafico iMRP para posteriormente procesarlo y así mostrar a los operadores en una pantalla las instrucciones a ejecutar. El proceso de abstracción de la ingeniería o receta consiste en crear un modelo iMRP en donde estén contenidas cada una de las operaciones que se realizan en el proceso de producción de un producto, para ello se crea la secuencia de las operaciones en el modelo iMRP con base en cada una de las fases de la receta. En la figura 5.11 se muestra un modelo iMRP creado con base en una receta.

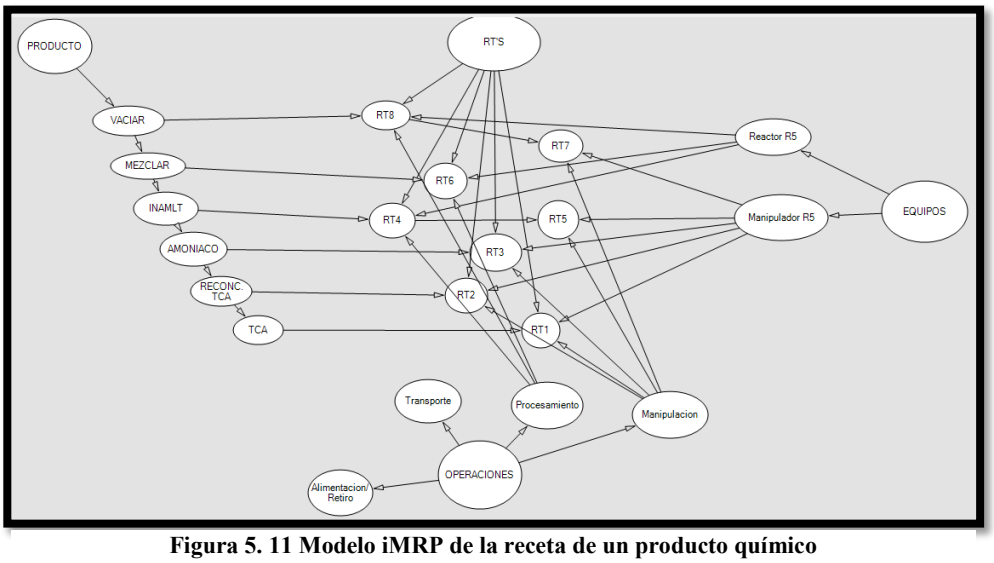

Figura 5. 11 Modelo iMRP de la receta de un producto químico

El proceso de validación del proyecto de tesis en la empresa de productos químicos consistió en instalar la computadora donde se encuentra el sistema lanzador de órdenes en una sala de control, por otra parte, el controlador arduino (botonera) y la computadora (Intel NUC) donde se encuentra la aplicación envoltura fueron instalados en un lugar cercano al piso de producción ya que en esta computadora está conectada a una pantalla en la cual se muestran las operaciones a los operadores.

Previo a la operación del sistema se adquirió una receta de un producto químico, dicha receta fue abstraída en un modelo iMRP de tal manera que cada una de las fases de la misma representa un recurso tiempo del modelo iMRP, por lo tanto cada fase tiene una relación con un equipo y una operación.

Además, se creó un archivo de texto el cual es leído y procesado por los objetos envoltura, dicho archivo incluye los pasos para realizar una ingeniería (receta) de un producto, en los cuales se están contenidas características como origen, destino, temperatura, pH, tiempo, etc.

Las pruebas de validación del sistema se realizaron en paralelo a los procesos que se realizan para la realización de una receta, en la figura 5.12 se muestra el diagrama en el cual se presentan los módulos y el funcionamiento del sistema lanzador de órdenes.

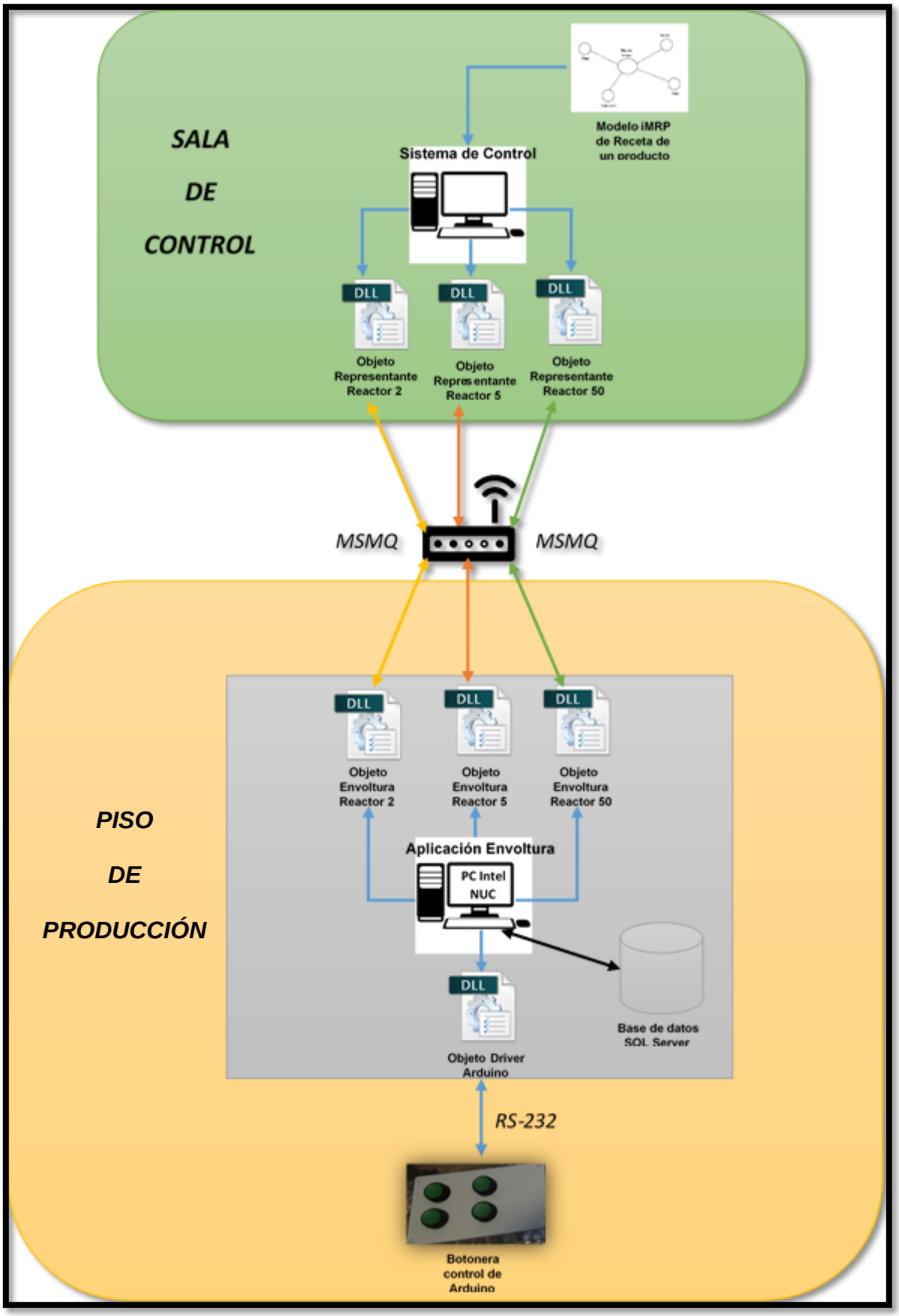

Figura 5. 12 Módulos del sistema instalado en empresa de productos químicos

El proceso de operación consistió en utilizar el sistema informático lanzador de órdenes de (Gonzalez Hidalgo, 2017) mostrado en la figura 5.13, el cual lee el archivo XML en el cual está contenida la información de la receta de un producto químico, así como los equipos utilizados y el orden de las operaciones. Con base en la información del archivo, el sistema crea los objetos representantes los cuales son utilizados para establecer la comunicación con los objetos envoltura de los distintos equipos procesadores (reactores) y manipuladores (bombas).

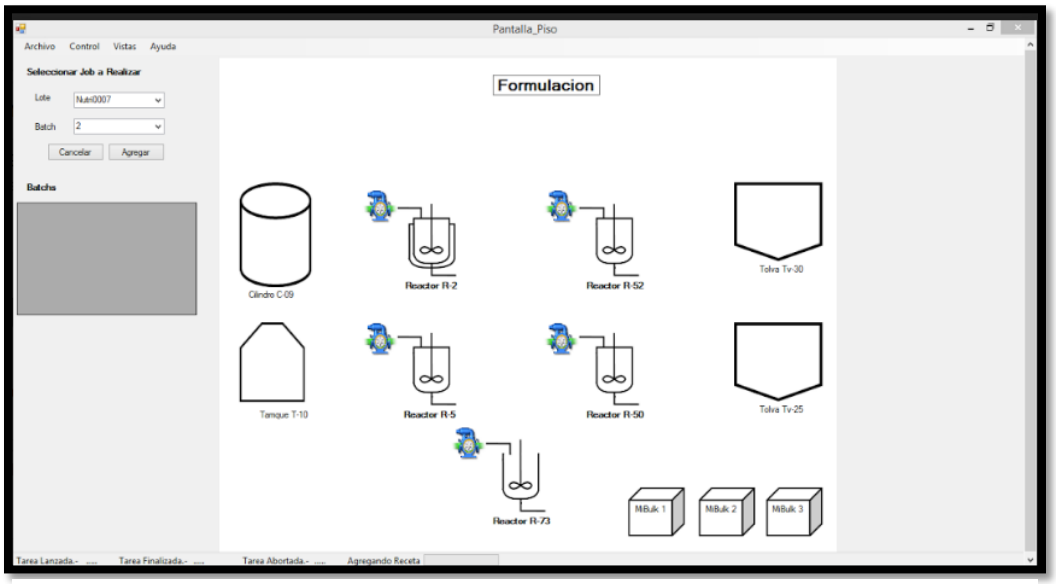

Figura 5. 13 Sistema de control de (Gonzalez Hidalgo, 2017).

Otra de las funciones que presenta el sistema lanzador (Gonzalez Hidalgo, 2017) es mostrar las actividades o fases de la receta en un diagrama de Gantt en Microsoft Project, tal como se muestra en la figura 5.14. Dicho diagrama es creado con base en el modelo iMRP, en él se ve reflejado el orden, la duración aproximada y el equipo responsable de realizar cada una de las fases, lo cual es de gran utilidad ya que de esta manera se puede monitorear en tiempo real el proceso de las recetas, además en dicha pantalla se puede ver reflejado cuando hay un paro de emergencia o cuando una tarea fue terminada, aportando un mejor control de los procesos.

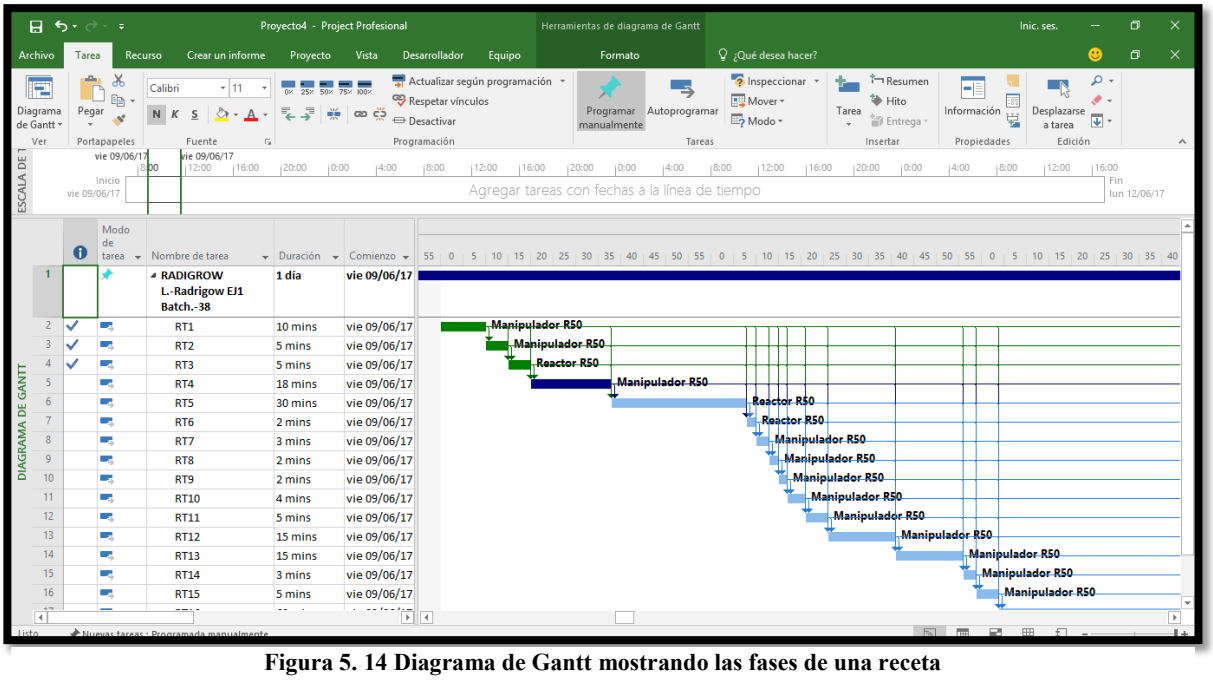

Figura 5. 14 Diagrama de Gantt mostrando las fases de una receta

Por otra parte, al ejecutar la aplicación envoltura la cual está ubicada en una computadora distinta (PC Intel NUC) donde se encuentra el sistema de control, esta lee un archivo XML en el cual se encuentra la información de los equipos del piso de producción y con base en el archivo crea dinámicamente al varios objetos envoltura, por lo tanto, todos los objetos envoltura se encuentran en la misma PC y además crea un solo objeto propulsor (driver), debido a que todas los objetos envoltura se comunican por el mismo medio físico (RS-232) con el controlador Arduino (Botonera), se requirió únicamente un objeto propulsor.

Después de iniciar ambas aplicaciones, el sistema lanzador envía a través de los objetos representantes el nombre y la ubicación del archivo de texto al objeto envoltura correspondiente, cuando éste recibe el mensaje, lee el archivo de texto en el cual se encuentran cada una de las fases e instrucciones de la receta (a realizar por el operador). Dichas instrucciones son desplegadas en la pantalla de la aplicación envoltura para que el operador las realice, en la pantalla se despliega una ventana que contienen el nombre del equipo y del operador, así como también la fase a realizar, y algunas otras características como la temperatura, el pH, etc. tal como se muestra en la figura 5.15.

Cuando una instrucción es recibida por el objeto envoltura, este inserta en la base de datos la hora y la fecha del inicio de operación del equipo.

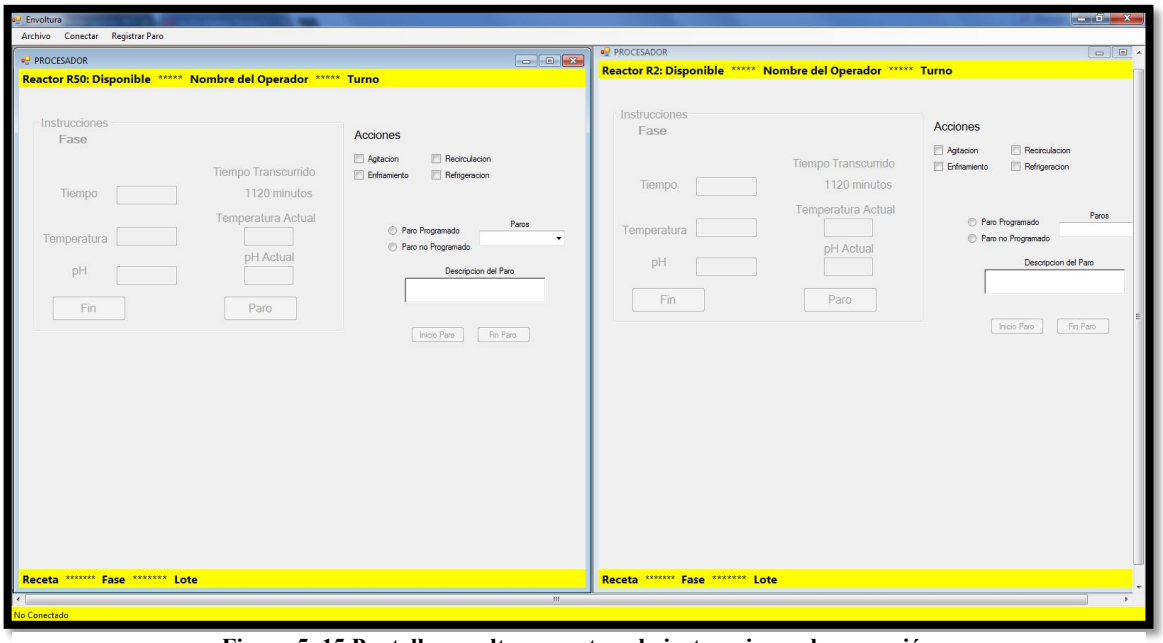

Figura 5. 15 Pantalla envoltura mostrando instrucciones de operación

Por otra parte, a causa de que los equipos no cuentan con un controlador automático, fue necesario crear control genérico utilizando un arduino para poder retroalimentar el evento de fin de operación o el paro de emergencia de todos los equipos de manera manual el cual es controlado a través de botones que activan las entradas del mismo, tal como se muestra en la figura 5.16; Algunas entradas tiene relación con los equipos y otras indican la acción a realizar (fin de operación, paro de emergencia), por lo tanto, el funcionamiento del controlador consiste en pulsar dos botones, el primer botón indica el equipo y el segundo botón la acción a realizar, luego de pulsar la combinación de botones se envía un comando donde está contenido el nombre del equipo y la acción correspondiente al objeto driver de la aplicación envoltura a través del protocolo RS-232 y el driver re-envía dichas instrucciones al objeto envoltura y este registra en la base de datos la fecha y hora del fin de operación o del inicio del paro de emergencia y envía la instrucción de fin de operación al objeto representante.

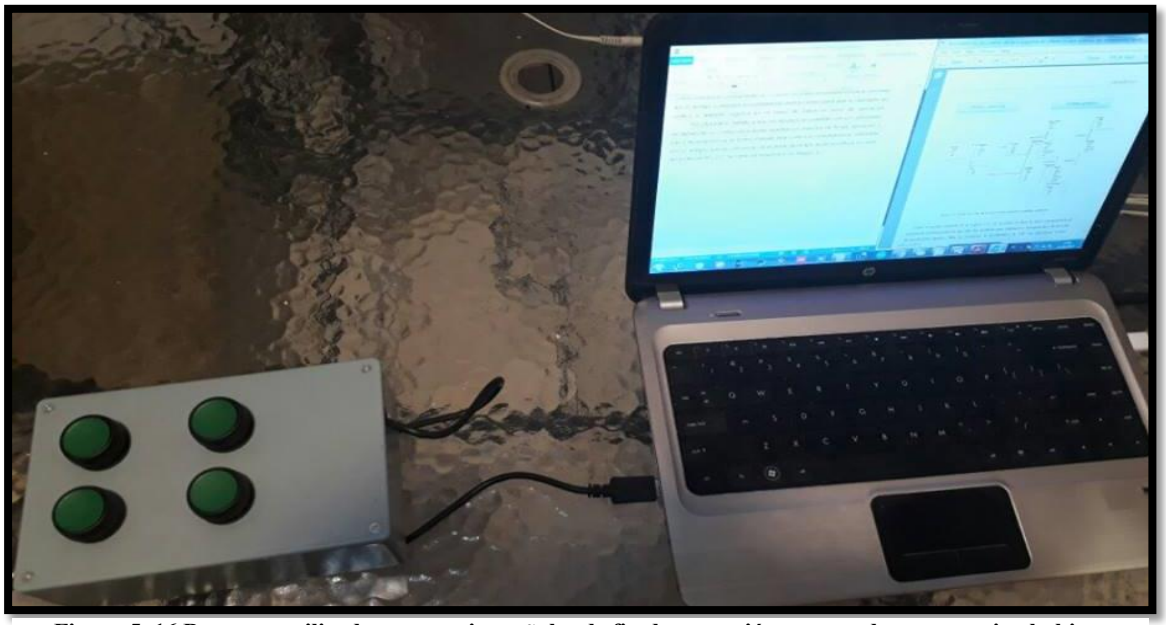

Figura 5. 16 Botonera utilizada para enviar señales de fin de operación o paros de emergencia al objeto propulsor

5.3 APLICACIÓN DE REPORTES Y CÁLCULO DE OEE EN UN PISO REAL DE PRODUCCIÓN.

Otro de los requerimientos de la empresa es medir el desempeño de los equipos y generar reportes con la información almacenada en la base de datos. Para validar el funcionamiento de esta aplicación se almacenaron varios registros en las tablas de la base de datos y con base en esos datos se generan reportes, los cuales muestran graficas con información sobre la duración de tiempos muertos de los equipos, las ingenierías, operadores o catálogos y además tiene la función de filtrar datos para obtener información más exacta según sea requerida, figura 5.17.

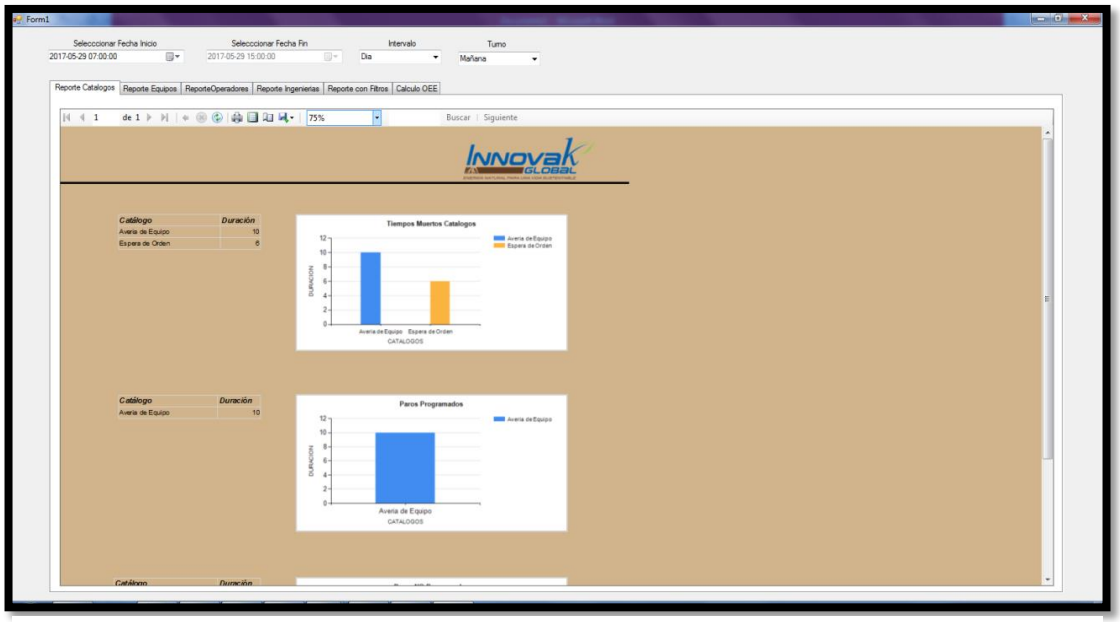

Figura 5. 17 Aplicación para calcular OEE y generar reportes

Por otra parte dentro de la aplicación hay un módulo que permite realizar el cálculo del OEE para los equipos existentes en el piso de producción, para realizar dicho cálculo es necesario seleccionar la fecha de inicio, el intervalo de tiempo (día, semana, mes, etc.), el turno y el equipo al cual se le desea calcular el rendimiento.

Al realizar el cálculo, en la pantalla se despliegan las gráficas que muestran la relación de las pérdidas de disponibilidad, velocidad y calidad y finalmente muestra el valor del OEE obtenido, tal como se muestra en la figura 5.18.

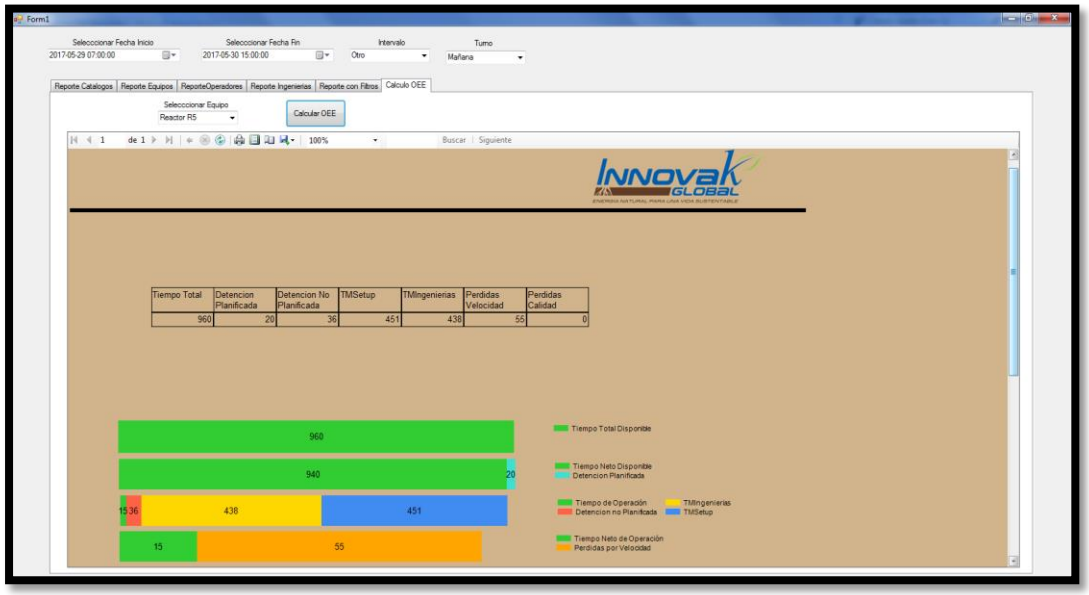

Figura 5. 18 Pantalla del cálculo del OEE

5.4 ANÁLISIS DE RESULTADOS.

Con base en los ejercicios de validación descritos anteriormente, se obtuvieron varios resultados, entre los que podemos mencionar que la aplicación de la arquitectura de referencia ArquiTAM permite crear un sistema de control modular y flexible capaz de integrar nuevos equipos a un piso de producción sin requerir esfuerzos de re-programación del sistema de control.

Uno de los resultados alcanzados fue que al cambiar de sistema de producción de un célula robotizada que simula un sistema de producción discreta a un sistema fabricación por lotes no fue necesario modificar el código fuente del sistema de control para integrar nuevos equipos de producción sin importar el tipo y las características de los mismos como medio físico de comunicación aceptado o lenguaje de programación utilizado. Además, debido a que se utilizó un solo protocolo de comunicación (Servicio de colas de mensajes de Windows (MSMQ)) para establecer la comunicación entre los objetos representantes y el objeto envoltura, sólo fue necesario configurar una sola envoltura.

Además el esquema de acoplamiento débil fue utilizado para lograr la integración entre los módulos del sistema (objeto representante, envoltura y propulsor) para ofrecer autonomía al sistema de control, en este caso se utilizó el servicio de colas de mensajes de Windows MSMQ. También, el concepto de operación dirigida por modelo es una característica que tiene el sistema de interpretar un modelo gráfico y con base en la información del modelo lanzar órdenes de operación a los equipos.

El sistema de rastreabilidad, resulta de gran utilidad en la parte administrativa de la empresa, dicho sistema permite el cálculo del OEE, es decir, calcula la eficiencia de los equipos en un determinado lapso de tiempo, por otra parte genera reportes los cuales permiten analizar las principales fallas de los equipos, así como detectar cuales equipos tienen más incidencias de fallos, y otra información relevante del piso de producción.

CAPÍTULO 6. CONCLUSIONES Y RECOMENDACIONES

El sistema de control realizado con base en los lineamientos de la arquitectura de referencia ArquiTAM permitió el desarrollo de un sistema modular flexible capaz de adaptarse fácilmente a los posibles cambios que pueden surgir en un piso de producción eliminando la necesidad de modificar el código fuente del sistema principal.

Al realizar pruebas de validación se encontró que los módulos del sistema son reusables, es decir, puede ser implementado tanto en un piso de producción discreta como en un ambiente de producción por lotes, además, debido a que el sistema realiza la integración de equipos y el lanzamiento de órdenes con base en acoplamiento débil y con base en operación dirigida por modelo (modelo grafico iMRP) esto le da la capacidad de integrar nuevos equipos y procesos de producción sin algún esfuerzo extra de programación del código principal, reduciendo todo el trabajo a modificar el modelo gráfico.

Por otra parte, además de integrar diferentes equipos y procesos, el sistema es capaz de incorporar funciones (flexibilidad), como fue el caso de la función de monitorear el desempeño de los equipos de un piso de producción, función que fue agregada posteriormente a la implementación del sistema de control; tal función ayuda a la parte administrativa de la empresa en la toma de decisiones para mejorar los procesos.

Al implementar el sistema en un ambiente real de producción surgieron nuevos requerimientos, sin embargo, a pesar de ello no fue necesaria la modificación de los módulos principales (sistema de control y envoltura), sólo fue necesaria la creación de nuevos objetos driver para poder establecer la comunicación con los distintos equipos.

Trabajo Futuro

Actualmente el sistema ha sido validado en una célula de manufactura que emula un entorno de producción discreta y también en un piso de producción por lotes real, sin embargo, existen otras áreas de oportunidad donde este tipo de sistemas puede ser aplicable, como el área de fabricación continua. Si bien se han hecho las adaptaciones para utilizar los modelos iMRP en el área de producción discreta y por lotes, también podría trabajarse en hacer las adaptaciones para modelar los procesos continuos con iMRP y aplicar el sistema de control y de medición de OEE en esa área.

Referencias

Acosta , J., & Sastrón F. (2007). Production shop floor system modeling with reusability focus. *IFAC Conference on Cost Effective Automation in Networked Product Development and Manufacturing.* Monterrey, MX.

Acosta C, J., Medrano O., M., & Fuentes O., H. (2012). *Desarrollo del Sistema Informático para manejo del Sistema Estadístico y rastreabilidad de resultados de prueba en el proceso de fabricación de bombas en la empresa Delphi.* Convenio ITCH-AID-Delphi, Chihuahua.

Acosta Cano de Los Rios, J. E. (2016). Esquema de Referencia para Acoplamiento Débil en Sistema Informático y Equipo de Producción. *Universidad Politecnica de Madrid*.

Acosta Cano, F., & Acosta Cano, J. (2013). *Modelado Orientado a Objetos del Proceso de Administración de la Obra Pública.* Chihuahua: DGEST.

Acosta Cano, J. (2014). *Sistematización de las funciones de programación de órdenes de trabajo y la coordinación de flujo y procesamiento de fase (SCFP) en un sistema de producción por lotes.* Empresa Productos Químicos de Chihuahua e ITCH, Chihuahua.

Acosta, C. , J., & Sastron, B. (2006). Schematic Architecture: Reference Architecture/Frameworks/Particular Models for the Shop Floor Environment. *IECON 2006-32nd Annual Conference* (págs. 4563-4568). IEEE Industrial Electronics.

Aguiar, A. J., Villani, E., & Junqueira, F. (2011). Petri Coloreadas redes y simulación gráfica para la validación de una célula robótica en la industria aeronáutica. 929-941.

Ames, V., Gililland, J., Konopka, J., & Schnabl, R. (1995). *Semiconductor Manufacturing Productivity Overall Equipment Effectiveness (OEE) Guidebook Revision 1.0.* SEMATECH.

Bauer, N., Huuck, R., Lukoschus, B., & Engell, S. (2004). *An unifying semantics for sequential function charts.* New York, USA: Lecture Notes in Computer Science.

Chen , S.-M., Jyh-Sheng , k., & Chang, J.-F. (2002). Representación del conocimiento mediante redes de Petri difusos. *IEEE*, 311-319.

Dennis, B. (2006). *Design Patterns for Flexible Manufacturing.* ISA.

Eduardo. (Abril de 2012). Obtenido de http://eduardo-ingenieriadesoftware.blogspot.mx/

García Moreno, E. (1999). *Automatización de procesos industriales: robótica y automática.* Valencia: Ed. Universidad Politécnica.

Gómez García, M. S. (2012). *Esquema de Referencia para Sistemas de Rastreabilidad basado en ArquiTAM.* Instituto Tecnológico de Chihuahua. Chihuahua: DEPI.

Gonzalez Hidalgo, M. A. (2017). Aplicación de la técnica iMRP en la operación dirigida por modelo en un área de producción por lotes. Chihuahua, Chihuahua, México.

Granda, M. (2012). *EXTENSIONES DE LAS REDES DE PETRI: TEMPORIZACION*. Obtenido de http://www.ctr.unican.es/asignaturas/mc_procon/doc/petri_4.pdf

Haines, L., & Evers, K. (1990). An IDEF0 representation of the instructional system development (ISD) process-why it works. *Aerospace and Electronics Conference, IEEE 1990 National* (págs. 806- 811). Papers.

Hansen, R. C. (2001). *Overall Equipment Effectiveness A Powerfull Production Maintenance Tool for Increased Profits.* New York: Industrial Press Inc.

Huayna, A. M., Cortez Vásquez, A., & Vega Huerta, H. (2009). Aplicación de las redes de Petri a la simulación discreta de sistemas. *Revista de Ingeniería de Sistemas e Informática vol. 6*, 1-10.

Jiang, J., Azzopardi, D., Holding, D., Carpenter, G. F., & J.S.Sagoo. (1996). Real-time synchronisation of multiaxis high-speed machines, from SFC specification to Petri net verification. *IEE Proc*.

Lange, C., Chaudron, M., & Muskens, J. (2006). In practice: UML software architecture and design description. *Software, IEEE , vol.23, no.2*, 40,46.

Lindemann, C. (1998). Performance Modelling with Deterministic and Stochastic Petri Nets. 1-12.

Liu , F., Blätke, M.-A., & Heiner , M. (2014). nets, Modelling and simulating reaction–diffusion systems using coloured Petri. *ELSEVIER*, 1-12.

Llorens, M., & Oliver, J. (2004). *Redes Reconfigurables Controladas por Marcado: Redes de Petri con Cambios Dinámicos Estructurales*. Obtenido de Redes de Petri con Cambios Dinámicos Estructurales: http://users.dsic.upv.es/grupos/elp/mllorens/jjc

Medina Marin , J. (2002). Red de Petri Coloreada Condicional (CCPN) y su Aplicación en Bases de Datos Activas.

Murata, T. (1989). Petri Nets: Properties, Analysis and Applications. *Proceedings of the IEEE*, 541-580.

Oechsner, R., Pfeffer, M., & Pfitzner, L. (2003). Overall equipment efficiency (OEE) to overall Fab effectiveness (OFE). *PERGAMON*.

Petri, C. (1962). Kommunikation mit Automaten. Schriften des Rheinisch-Westfälischen Institutes für Instrumentelle Mathematik an der Universität Bonn.

Pomorski, T. (1997). Managing Overall Equipment Effectiveness to Optimize Factory Performance. *IEE*.

Prado, C. (28 de Marzo de 2015). *Modelamiento de procesos utilizando la metodología IDEF0*. Obtenido de https://es.slideshare.net/carloslonkanprado/metodologia-idef0

Recuero, A., & Alvarez, M. (1997). APPLICATIONS OF PETRI NETS IN CONSTRUCTION.

René, D. (1995). Grafcet: A Powerful Tool for Specification of Logic Controllers. *IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY*, 253-267.

Sheu, D. D. (2006). Overall Input Efficiency and Total Equipment Efficiency. *IEEE*.

Singh, R., Shah, D. B., Gohil, A. M., & Shah, M. H. (2012). Overall Equipment Effectiveness(OEE) Calculation-Automation through Hardware&Software Development. *ELSEVIER*.

Vorne, I. (2016). *OEE.com oee made easy by verone*. Obtenido de http://www.oee.com/

Wightkin, N., Buy, U., & Darabi, H. (2011). Formal Modeling of Sequential Function Charts With Time Petri Nets. *IEEE*.

Wilfried , B., & Wolfgang , R. (2006). Carl Adam Petri and "Petri Nets". *Springer - Verlag*, 369-374.

Zurawski , R., & Zhou, M. (1994). Petri Nets and Industrial Applications: A Tutorial. *IEEE*.

Currículum Vitae

Mi nombre es Carlos Iván Ontiveros Roacho, nacido en H. del Parral, chihuahua el 15 de Mayo de 1993, tengo 24 años de edad, soltero.

Soy egresado del CBTIS 138 de Ciudad Jiménez, Chihuahua donde cursé la especialidad de Informática durante los años 2007 al 2010 obteniendo el título de técnico en informática.

Posteriormente estudie la carrera de Sistemas computacionales Hardware en la facultad de Ingeniería de la UACh, en la Ciudad de Chihuahua, Chihuahua durante los años 2010 al 2015 obteniendo el título de Ingeniero en Sistemas Computacionales Hardware.

Durante los años 2015 al 2017 curse la maestría en Ingeniería en Computación en la misma facultad de ingeniería de la UACh, especializado en el área de automatización y robótica obteniendo el grado de maestro en Ingeniería.

En el Año 2016 cursé el diplomado de inglés que se imparte en el centro de idiomas de la UACh, obteniendo el diploma de acreditación del curso.

En el año 2017 comencé a trabajar en la empresa de telefonía Teleperformance Inc. en la Ciudad de Chihuahua durante un periodo de tres meses.

Domicilio Permanente: Campo del Manzanar 9710, Campo bello
Chihuahua, Chihuahua, C.P. 31124

Esta tesis/disertación fue mecanografiada por Carlos Iván Ontiveros Roacho