Trabajo de Fin de Grado
Grado en Ingeniería en Tecnologías Industriales

Digital Twin Systems Modelling to Improve Real Time Assets Operation and Maintenance.

(Modelado de sistemas gemelos digitales para la mejora de operaciones y mantenimiento de activos en tiempo real.)

Autor: Adolfo Crespo del Castillo

Tutor: Adolfo Crespo Márquez

Dep. Organización Industrial y Gestión de Empresas I
Escuela Técnica Superior de Ingeniería

Sevilla, 2018

Trabajo de Fin de Grado

Grado en Ingeniería de Tecnologías Industriales

Digital Twin Systems Modelling to Improve Real Time Assets Operation and Maintenance.

Autor:

Adolfo Crespo del Castillo

Tutor:

Adolfo Crespo Márquez

Dep. De Organización Industrial y Gestión de Empresas I

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2018

Trabajo Fin de Grado: Digital Twin Systems Modelling to Improve Real Time Assets Operation and Maintenance.

Autor: Adolfo Crespo del Castillo

Tutor: Adolfo Crespo Márquez

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2018

El Secretario del Tribunal

AGRADECIMIENTOS

En primer lugar quiero agradecer al professor Marco Macchi del Politecnico di Milano, por ofrecerme la oportunidad de realizar la estancia en la que se ha desarrollado esta tesis, así como por orientar mi trabajo y ofrecer una visión desde su conocimiento aportando inumerables consejos. Agradecer también a Edoardo Sottoriva, mi orientador y supervisor en el ámbito del Proyecto MAYA de digitalización de la industria, por realizar un seguimiento y aportaciones de gran valor, así como por su dedicación. En este primer apartado quiero agradecer especialmente a mi tutor Prof. Adolfo Crespo Márquez por su trabajo y dedicación a esta tesis, guiando y siguiendo cada paso con gran dedicación; además de ser un apoyo constante en el día a día.

En segundo lugar quiero agradecer al equipo del Dipartimento di Ingegneria Gestionale e Industry 4.0 Lab por hacerme sertir integrado y con un constante apoyo durante esta estancia, de manera que la realización de este trabajo ha sido lo más cómoda posible y me he sentido muy a gusto y arropado en todo momento. Debo hacer especial mención a mis compañeros Felipe J.Repetto y Francisco Tapia, por ser mi mayor apoyo en el PoliMi estos meses, y ofrecer su experiencia y conversación, de las que tanto he aprendido.

Debo agradecer también su comprensión y solidaridad, así como infinitas historias, a mis compañeros y amigos durante este periodo en Milán, especialmente por su ayuda a mi *coinquilino* Andrés García-Baquero León.

Por último y no menos importante, agradecer infinitamente a mi familia y amigos por estar a mi lado y aguantarme con cariño durante los años de estudio de esta ingeniería.

CONTENT

RESUMEN DEL TRABAJO EN CASTELLANO 11

Motivación y objetivos 11

Conceptos centrales y trasfondo tecnológico 12

El concepto Gemelo digital (Digital Twin, DT) 13

Técnicas y herramientas utilizadas para el desarrollo del trabajo 16

Estructura del proceso de trabajo con IDEF 18

Desarrollo de los modelos del trabajo 20

Resultados del trabajo 26

Conclusiones y futuros trabajos 27

CHAPTER 1. INTRODUCTION AND OBJECTIVES 29

1.1. Introduction 29

1.2. Thesis Motivation 30

1.3. Thesis Background 31

1.4. Thesis Objetives 31

1.5. Summary 32

CHAPTER 2. TECHNOLOGICAL BACKGROUND 34

2.1. Industry 4.0 34

2.1.1. Introduction to the concept 34

2.1.2. Pillars 34

2.1.3. Industry 4.0 on simulation 36

2.2. The Framework. The MAYA Project in PoliMI 37

2.2.1. Cyber-Phisical System 38

2.2.2. Synchronization of the digital and real Factory 39

2.2.3. **Current simulation practice in manufacturing** 40

2.2.4. Multidisciplinary integrated simulation and modelling 43

2.3. The Digital Twin Concept 43

2.3.1. Introduction 43

2.3.2. Literature Review 44

2.3.3. When was the concept of "Digital Twin" brought up? 45

2.3.4. What is "Digital Twin"? 45

2.3.5. Are there any synonyms of "Digital Twin"? 47
2.3.6. What are the key enabling technologies for the "Digital Twin"? 49
2.3.7. Benefits of the Digital Twin 52
2.4. Industry 4.0 Lab at PoliMI 53
2.4.1. Lab capabilities 53
2.4.2. The Product 55
2.4.3. OPC Unified Architecture 55
CHAPTER 3. TECHNIQUES, TOOLS AND METHODS BACKGROUND 57
3.1. IDEF() For representation for flow charts and process 57
3.2. Matlab/Simulink OPC Toolbox for Data Acquisition 59
3.2.1. Communicating with PLCs. 59
3.2.2. Matlab level 2 S-Function for Data Adquisition 60
3.3. Asset Health Index basis for Machine Condition Assessment 61
3.3.1. Background and basic definitions. 62
3.3.2. Data requirements. 64
3.3.3. Procedure. 64
3.4. Artificial Neural Networks (ANN) to Model Complex Behavior 66
3.4.1 The forwards pass 69
3.4.2 Calculating the Total Error 70
3.4.3 The Backwards Pass 70
3.5. Introduction to Continuous Time Dynamic Simulation 73
3.5.1. Stock and flow diagram 74
CHAPTER 4. WORK PROCESS DESIGN USING IDEF() 76
4.1 The IDEFØ Diagram of the process 76
4.2 IDEF1 Diagram 77
CHAPTER 5. MODELS DEVELOPMENT 82
5.1 Convert plant PLC signals into model variables 82
5.2 Single Machine Model (Machine state determination) 82
5.2.1 Single Machine model on Energy consumption 86
5.3 Drilling Machine Condition Modelling 87
5.3.1. Model Notation and Equations 91
5.3.2. Model Adaptation to Matlab Simulink 94
5.4 Machine Artificial Neural Network (ANN) 95
5.4.1 A Continuous Time Dynamic Simulation Model for the ANN 95

5.4.2 The ANN Dynamic Simulation Model....96
5.4.4 Notation of the variables....98
5.4.5 Equations....99
5.5. ANN Training....100
5.5.1. Calibration of the model parameters for ANN Training....100
5.5.2. Models Results and Validation....106
CHAPTER 6. RESULTS DISCUSSION....108
CHAPTER 7 CONCLUSIONS....112
REFERENCES....114
APPENDIX (MATBLAB/OPC UA/VENSIM CODES)....118
Matlab Simulink variables in real time....118
Level 2 S-Function of Drilling Module....118
Level 2 S-Function "DataTypeID"....119
Time Accumulator Matlab Function....120
Drilling Machine Condition Model Matlab Implementation....121
Matlab OPC UA Toolbox Functions....125
Vensim Simulation Model Code....126

RESUMEN DEL TRABAJO EN CASTELLANO

Motivación y objetivos

Este trabajo surge en el marco de una estancia en el Politécnico de Milán durante cuatro meses. Esta estancia se basa en el trabajo en el ámbito de la Industria 4.0 y la digitalización de la manufactura, y concretamente en el proyecto europeo MAYA. La posibilidad de realizar los experimentos con un equipo digitalizado, se da en el Industry 4.0 Lab del Politécnico de Milán, que ofrece un nivel puntero de desarrollo digital para el desarrollo del trabajo planteado.

En el sector industrial a día de hoy, se está gestando una reestructuración a todos los niveles, motivada por la llegada de la digitalización. Esto implica nuevas necesidades, formación renovada para los profesionales, y sobre todo la concienciación sobre una nueva revolución industrial en ciernes. Mediante este proceso de cambio la fábrica se enfoca a una fábrica digitalizada, que puede satisfacer mejor las necesidades del cliente (menos espera, y más personalización), y hacer un mejor aprovechamiento de los recursos.

Dentro de este ámbito, este trabajo se enfoca en el desarrollo de modelos digitales para el estudio de sistemas físicos, que permiten un control a tiempo real del funcionamiento de los activos y mediante esto asentar una mejor toma de decisiones de gestión y mantenimiento a tiempo real.

En la realización de este trabajo se tiene en consideración que los modelos gemelos digitales (DT) se encuentran en la infancia de su desarrollo, y que están sujetos a diferentes interpretaciones. Pero siendo consciente de esto, se ha desarrollado el estudio con la vista puesta en la adaptación digital de la fábrica y la aplicación de procedimientos avanzados de gestión de activos a tiempo real de proceso.

En consecuencia a todo lo mencionado anteriormente, en este trabajo se realiza una aproximación a los conceptos de Industria 4.0, Big Data, Internet of Things, y la digitalización. Una vez realizada una introducción al estado del arte y al proyecto MAYA, se presenta una búsqueda en literatura científica del concepto de gemelo digital (DT).

El objetivo de este trabajo se basa en la implementación de modelos gemelos digitales de sistemas físicos, después una sincronización que permita monitorizar el trabajo a tiempo real. Llegados al punto de poder representar un activo físico de manera fiel a nivel digital, el objetivo evoluciona a la aplicación de modelos para el control de la salud del activo en estudio. Este ha sido el objetivo de este trabajo, ofrecer una metodología completa, que englobe todos los procesos de gestión, que aporte herramientas y modelos, que proporcionen un respaldo y un soporte objetivo a los protocolos de acción a la hora de la toma de decisiones.

Conceptos centrales y trasfondo tecnológico

Industria 4.0: El concepto Industria 4.0 corresponde a una nueva manera de organizar los medios de producción. El objetivo que pretende alcanzarse es la puesta en marcha de un gran número de fábricas digitalizadas o "inteligentes" capaces de una mayor adaptabilidad a las necesidades y a los procesos de producción, así como a una asignación más eficiente de los recursos, abriendo así la vía a una nueva revolución industrial o cuarta revolución industrial. En esta nueva reorganización el software desplazará a la máquina en su importancia en el equipo productivo, el producto guardará memoria de su ciclo de vida, y los autómatas dejarán estar encerrados e interactuarán con los humanos.

Este concepto de nueva estructuración industrial o Industria 4.0, fue manejado por primera vez en la Feria de Hanover (salón de la tecnología industrial) en el año 2011. Y en la misma feria pero en el año 2013, un informe detallando este concepto y sus implicaciones, también fue presentado y defendido por un selecto grupo de trabajo e investigación.

Proyecto MAYA: El proyecto europeo de investigación MAYA (MultidisciplinArY integrated simulAtion and forecasting tools, empowered by digital continuity and continuous real world synchronization, towards reduced time to production and optimization) es uno de los numerosos proyectos existentes para desarrollar el concepto de modelos CPS en el ámbito de la Industria 4.0, y financiado por algunas de las mayores empresas europeas, y punteras en la digitalización de la industria.

Este proyecto tiene tres objetivos básicos:

- Garantizar una transmisión de información relevante a lo largo del ciclo de vida de la fábrica. Para ello crear un loop a lo largo de todo el ciclo de vida para conseguir ventajas económicas, operacionales, y de conocimiento.

- Empoderar la sincronización del mundo digital con la fábrica real que se encuentra en continuo cambio. El gemelo digital (DT) es capaz de reajustarse en tiempo real, y por lo tanto ser un espejo del mundo real. Esto tiene grandes ventajas en términos de monitorización, optimización, eficiencia y anticipación al fallo.

- Crear un entorno en el cual diferentes herramientas de simulación pueden estar integradas e interactuar, para crear vínculos que faciliten la inicialización y sincronización. Esto permitirá correr simulaciones multidisciplinarias e interconectadas para un intercambio de datos más fácil y relevante.

Sistemas Ciber-Fisicos (CPS): Denominamos sistemas ciberfísicos (o cyber physical systems - CPS) a sistemas basados en tecnologías software/hardware y de comunicaciones incorporadas en dispositivos donde se establece un lazo cerrado entre el proceso digital de datos y/o señales y el fenómeno físico bajo supervisión y actuación. Se trata de una evolución de los sistemas empotrados con conectividad local, y están llamados a revolucionar no solo procesos de manufactura, sino

también los mismos productos gracias a su potencial de creación de nuevos modelos de negocio basados en la personalización extrema. Se acuña el término CPS para describir sistemas empotrados y redes dedicadas a la sensorización y actuación sobre procesos físicos, donde el proceso físico afecta a su vez al procesado digital de datos y señales en un lazo cerrado.

Puede decirse que un sistema ciberfísico está compuesto de dos partes bien diferenciadas. En primer lugar el sistema físico real que realiza un determinado trabajo en el mundo real. En segundo la parte virtual, en la cual se encuentra el modelo gemelo digital (DT). Tiene una dimensión estática ("Data Model") que representa el sistema. Existe otra segunda que contiene los modelos de simulación multidisciplinaria y amplia el foco a la interconexión de los CPS (tanto en protocolos de comunicación orientados a la internet de las cosas como a plataformas de servicios) y la distribución de procesado digital en la nube.

El concepto Gemelo digital (Digital Twin, DT)

Una vez contextualizado el trabajo, y desarrollado el marco que engloba los conceptos centrales; es importante centrar esta parte en indagar en el concepto del gemelo digital (DT). En primer lugar surge en 2003 por Michael Grieves en la Universidad de Michigan aunque en ese momento la representación digital era relativamente nueva e inmadura. El concepto toma importancia en 2011 en Hannover al ser uno de los principales términos de la Industria 4.0, aunque ya había sido utilizado por la NASA años antes.

Una vez sabido su origen, el significado del concepto tiene muchas acepciones o interpretaciones pues no existe una oficial o formal. Por ello se proponen algunas que provienen de la literatura, y son las más representativas o las cuales pueden servir de confluencia para muchas.

- Un Gemelo Digital (DT) es una simulación multidisciplinaria integrada que utiliza modelos creados a partir de sensores, indicadores e historial para crear un modelo que represente como un espejo la realidad en un entorno virtual.

- Un Gemelo Digital (DT) es una representación del producto real con información desde el principio de la vida hasta su final.

- Un Gemelo Digital (DT) es un modelo con vida, de un sistema o activo, que continuamente se adapta a los cambios en el entorno o las operaciones, para ofrecer el mejor resultado favoreciendo la toma de decisiones.

Partiendo de estas definiciones y de la contextualización, debe tenerse en cuenta la tecnología que permite la implementación de este tipo de modelos o sistemas. Las herramientas o tecnologías principales que permiten esto son el Internet de las cosas y la Big Data. Estas tecnologías están integradas en los modelos de simulación pues permiten la sincronización y la conexión física-virtual mediante sensores y PLCs en el sistema (ver Figura e.1).

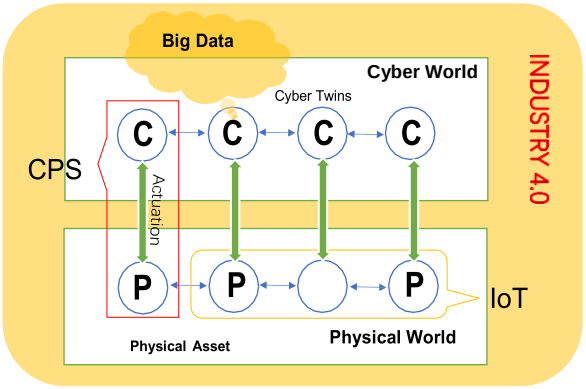


Figura e.1. Integración de tecnologías en CPS

Con esta conexión sincronizada el sistema físico envía la información de datos que será el input en el gemelo digital, después el modelo virtual monitoriza y trabaja de manera paralela y en las mismas condiciones. Una vez utilizado el modelo para la toma de decisiones y el estudio, existe el paso final de la realimentación que implica el envío de datos de control por parte del modelo virtual al real. Con lo cual el modelo virtual gemelo puede utilizarse como herramienta de monitorización, de toma de decisiones, o incluso como prototipo para posterior desarrollo de sistemas físicos.

Para implementar el DT es necesario la tecnología que habilita a este:

- *Big Data:* Existen múltiples definiciones del término Big Data, casi todas ellas coinciden en considerar Big Data como aquellas grandes cantidades de datos que no pueden ser procesadas de forma tradicional. Datos estructurados son aquellos que se gestionan en una base de datos relacional bajo un modelo de datos. Los datos no estructurados son aquellos que han de ser procesados para poder ser introducidos en una base de datos relacional, ejemplos de esto son imágenes, ficheros de texto, mails, vídeos, clips de audio, etc. En la cuarta revolución industrial esperamos grandes flujos de datos los tres pilares clave:

    - Sistemas Ciber-Físicos
    - Internet de las cosas
    - Internet de los Servicios.

    Big Data en términos de negocio significa nuevas oportunidades basadas en la toma de decisiones sobre grandes cantidades de datos heterogéneos, que van a permitir la optimización de los procesos, del servicio postventa y del mantenimiento.

- *Internet of Things (Internet de las cosas):* Actualmente, el término Internet de las cosas se usa con una denotación de conexión avanzada de dispositivos, sistemas y servicios que va más allá del tradicional M2M (máquina a máquina) y cubre una amplia variedad de protocolos, dominios y aplicaciones. Es un concepto que se refiere a la interconexión digital existente a día de hoy, de objetos cotidianos con Internet. Alternativamente, Internet de las cosas es la conexión de Internet con más “cosas u objetos” que personas. También se suele conocer como Internet de todas las cosas o Internet en las cosas.

Industria 4.0 Lab del Politécnico de Milán

Realizar el modelo de simulación del DT no habría tenido sentido sin tener la opción de validar lo programado. En este punto entra el Laboratorio del Politécnico de Milán, que permite debido a sus avanzados dispositivos la opción de trabajar con el DT a tiempo real y de manera empírica comprobar la validez del modelado.

El Laboratorio de Industria 4.0 está financiado y desarrollado por el departamento de Management and Industrial Engineering (Dipartimento di Ingegneria Gestionale) del Politécnico de Milán así como por FESTO y SIEMENS. Con el objetivo de crear un lugar de desarrollo e innovación en un entorno real. Puecden identificarse tres objetivos principales: Educación y entrenamiento, comunicación y consultoría, y proyectos de investigación. A nivel tecnológico está provisto de potentes herramientas para demostrar de manera académica:

- *CPS (Cyber-physical system)*: Permite una red de trabajo inteligente entre personas, productos, y recursos de producción. Es un enlace real entre el mundo real y el virtual.
- *RFID (Radio Frecuency Identification)*: Utiliza la transmisión por radiofrecuencia para leer o escribir datos entre dispositivos sin necesidad de contacto.
- *NFC (Near Field Communication)*: Permite la comunicación entre dos elementos situado cercanos.
- *Cloud Technology*: Guarda sensores y datos en la nube, y analiza Big Data.

Esta tecnología ha permitido construir una línea de ensamblaje y montaje de un teléfono móvil. Compuesta por siete estaciones cuyos componentes básicos son sus respectivos sensores, PLC, y su módulo propio de trabajo con una interfaz para su control. Las cintas transportan los pallets entres las estaciones, entre las cuales no existe buffer. Todo esto está conectado a dos ordenadores, de los cuales uno contiene el Manufacturing Execution System (MES) para generar y controlar la producción; y el otro un software de monitorización del consumo de energía. Todo esto a su vez está conectado a la red y cada dispositivo tiene asignada una dirección IP. Las estaciones del laboratorio son (Figura e.2):

1- *Manual Module*: El producto acabado está listo para ser retirado, o se carga el pallet vacío
2- *Base Magazine Module*: Se coloca una base en pallet vacío.
3- *Drilling Module*: Se realiza el taladrado
4- *Robot Cell*: E la única CP-Factory, una PCB(Printed Circuit Board) se coloca en la base y los elementos internos del móvil sobre esta.
5- *Visual Inspection Module*: Comprueba con un sensor óptico la correcta colocación de los elementos.
6- *Cover Magazine Module*: Se coloca la tapa del móvil .
7- *Press Module*: Se presiona la tapa para cerrar el móvil.

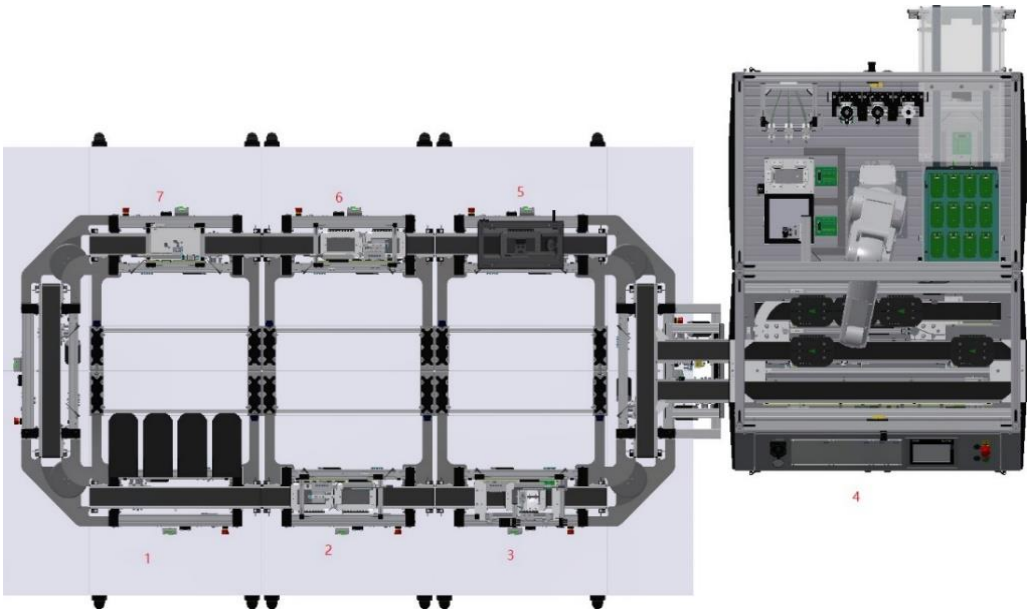


Figura e.2. Representación del Lab Industria 4.0 del PoliMi

Técnicas y herramientas utilizadas para el desarrollo del trabajo

Metodología IDEF: Para la representación del proceso del trabajo, se ha optado por un lenguaje intuitivo, claro y visual, que permite al usuario la comprensión del proceso con varios niveles de detalle. El lenguaje escogido es IDEF() (Integrated Definition for Function Modeling).

Permite la representación multinivel del proceso. En este caso se ha escogido desarrollar los niveles IDEFØ, e IDEF 1. Representa cada función, con sus entradas y salidas, de manera que las conexiones quedan claras; además, añade los mecanismos y herramientas necesarios (entrando desde abajo); así como las restricciones y controles (entrando desde arriba) como observamos en la siguiente imagen.

En el nivel IDEFØ se presenta la metodología global, con el máximo nivel de agregación (Figura e.3). En IDEF1 cada uno de los procesos se subdivide con mayor nivel de detalle y menor nivel de agregación.

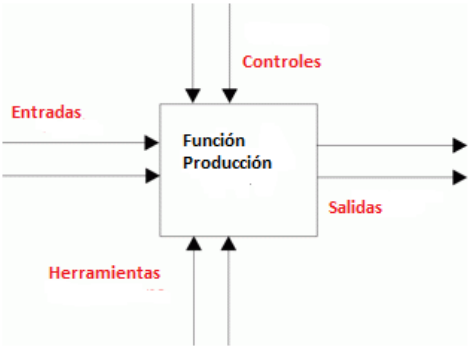


Figura e.3. Representación procesos con IDEF()

OPC UA Expert para la conexión con los PLC

Para la adquisición de datos el laboratorio adopta OPC Unified Architecture (OPC UA), un protocolo de comunicación Machine to Machine (M2M) desarrollado por OPC Foundation. Es una herramienta importante para la conexión para la extracción de data y comunicación. Con este protocolo además, se habilita a herramientas como Matalb y Simulink a conectarse al servidor mediante la red y el propio puerto de cada máquina.

Level 2 Matlab S-Function

Una vez conocidas las funciones necesarias en Matlab, es necesario utilizar el progama UAExpert para conectar con el servidor de PLC deseado mediante la red. Una vez conectado nuestro modelo y conocidas las funciones, es necesario desarrollar una función de Matlab que permita extraer los outputs deseados del servidor OPC UA, y esta se llama Level 2 Matlab S-Function.

Esta función define las características del output deseado del PLC, y hace que el método de callback se actualice en el tiempo para tener los datos a tiempo real. Esto se consigue mediante las funciones opcua() anteriormente mencionadas, y se refiere a un determinado puerto, con un determinado parámetro y estado; y el comportamiento de este. La correcta programación de esta función es crítica pues es tener los datos deseados a tiempo real es una de las bases sobre las que posteriormente se va a edificar el DT.

Metodología AHI: Los indicadores de salud de activos, son una herramienta muy utilizada para el mantenimiento basado en la condición del activo. Este indicador suele ser un indicador anidado de varios diferentes y proporciona una monitorización de las capacidades del activo durante su funcionamiento.

El indicador AHI determina la salud del active como una agregación de resultados del criterio que modela la condición y el peso de cada uno de estos. Para realizar un indicador eficiente se debe utilizar una unión de los diferentes tipos de datos disponibles para identificar cualquier evento que afecte a la condición del activo.

ANN (Red Neuronal Artificial): Las redes neuronales son un modelo computacional basado en un gran conjunto de unidades neuronales simples (neuronas artificiales), de forma aproximadamente análoga al comportamiento observado en los axones de las neuronas en los cerebros biológicos. Cada unidad neuronal está conectada con muchas otras y los enlaces entre ellas pueden incrementar o inhibir el estado de activación de las neuronas adyacentes.

Las redes neuronales suelen consistir en varias capas o un diseño de cubo, y la ruta de la señal atraviesa de adelante hacia atrás. Propagación hacia atrás (Backpropagation) es donde se utiliza la estimulación hacia adelante o en el "frente" para restablecer los pesos de las unidades neuronales y esto a veces se realiza en combinación con una formación en la que se conoce el resultado correcto. Modelos de redes neuronales en la inteligencia artificial se refieren generalmente a las redes neuronales artificiales (RNA); estos son modelos matemáticos esencialmente simples que definen una función o una distribución. Pero a veces los modelos

también están íntimamente asociadas con un algoritmo de aprendizaje en particular o regla de aprendizaje. Esto se va a utilizar para establecer un modelo de red neuronal en el DT que sea un método inequívoco de detección del fallo partiendo de ciertos outputs e inputs, y la correcta calibración de la red.

Estructura del proceso de trabajo con IDEF

Antes de explicar el desarrollo del trabajo, se ha realizado una contextualización y definición de los términos. Una vez acabado esto se procede a explicar el grueso del trabajo. Se realiza una explicación general y luego se procede a profundizar en cada una de las partes. El trabajo se ha centrado en el desarrollo de un modelo de DT en Matlab Simulink con simulaciones a tiempo real de un activo. Tal y como muestra la Figura e.4 con un modelo IDEFØ, el desarrollo de esta tesis utiliza diferentes enfoques de la utilización del DT.

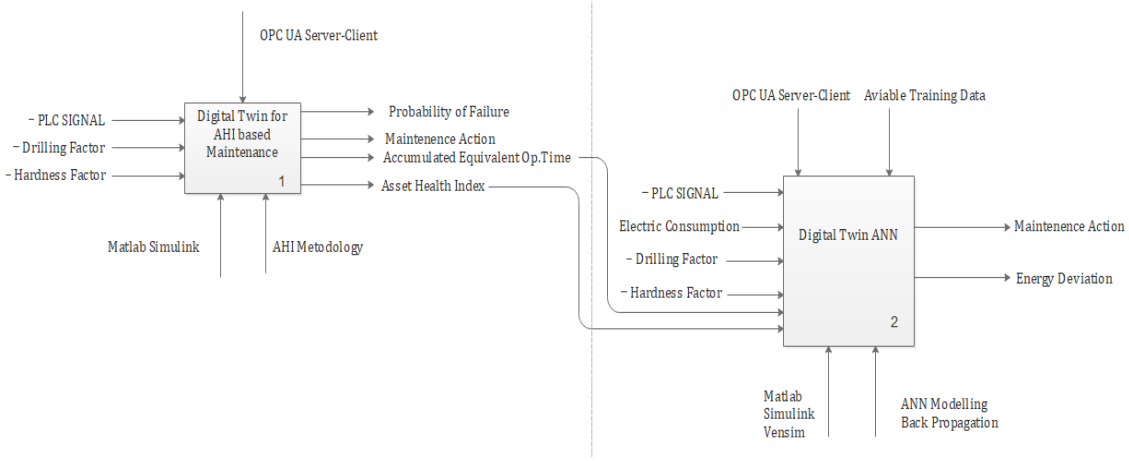


Figura e.4. Diagrama IDEFØ del proceso seguido en el trabajo

En primer lugar, en la rama superior *IDEFØ 1 box* se muestra la utilización del modelo DT del activo para convertir las señales del PLC en variables del modelo, monitorizar los estados por los que pasa la máquina, y generar posteriormente un histórico del tiempo que la máquina ha pasado en cada estado en el tiempo de la simulación (que es a tiempo real). Hasta este punto el DT solo funciona como un espejo digital que monitoriza y recoge datos a tiempo real (*IDEF1 1.2 box)*, y llegados a este punto se comienza el estudio y desarrollo de un modelo de simulación para obtener una medida de la condición o salud del activo, así como su probabilidad de fallo (*IDEF1 1.3 box)*. Esta sección modelo se integra en el DT, para tener integrado en el mismo una simulación multidisciplinaria a tiempo real del activo (AHI), lo cual permite realizar una toma de decisiones mediante que la aplicación de un control del mantenimiento basado en la condición del activo. Toda esta parte del proceso queda plasmada a mayor nivel de detalle en el diagrama IDEF1 de la Figura e.5.

Una vez explicado en profundidad el primer proceso del diagrama *IDEFØ,* procede desarrollar la aplicación del DT con ANN (Artificial Neural Network). En este trabajo se desarrolla el concepto de red neuronal así como un modelo, pues la red neuronal nos permite realizar una técnica de vigilancia del proceso de aparición y propagación del fallo, que se basa en desviaciones observadas en el

comportamiento del activo, respecto a su modo de comportamiento normal para una determinada condición operativa.

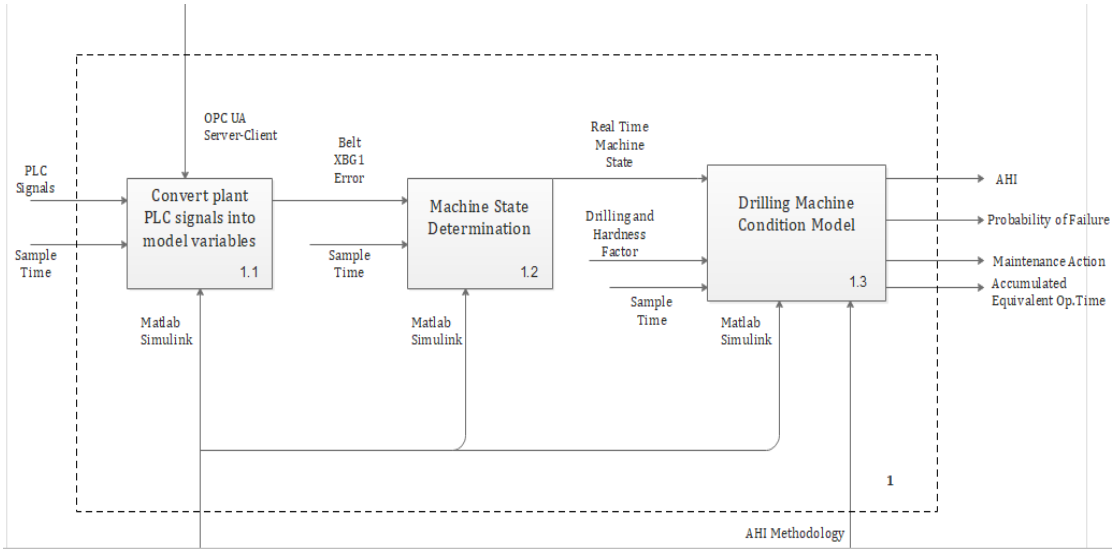


Figura e.5. Diagrama IDEF1 del proceso 1 del diagrama *IDEFØ*

Para ello se convierten de la misma manera que en el primer proceso, las señales del PLC en variables del modelo mediante la función o proceso 2.1 del diagrama IDEF1 de la Figura e.6. Después se realiza el entrenamiento de la red mediante un modelo de Vensim, para obtener una calibración de las variables del modelo que permita su posterior funcionamiento (IDEF1 2.2). Finalmente se hace la exposición de como esta red neuronal trabajaría y cual sería su función, pero sin llegar a implementarse en el DT de Matlab simulink, pues no existen valores reales de degradación, son simulados.

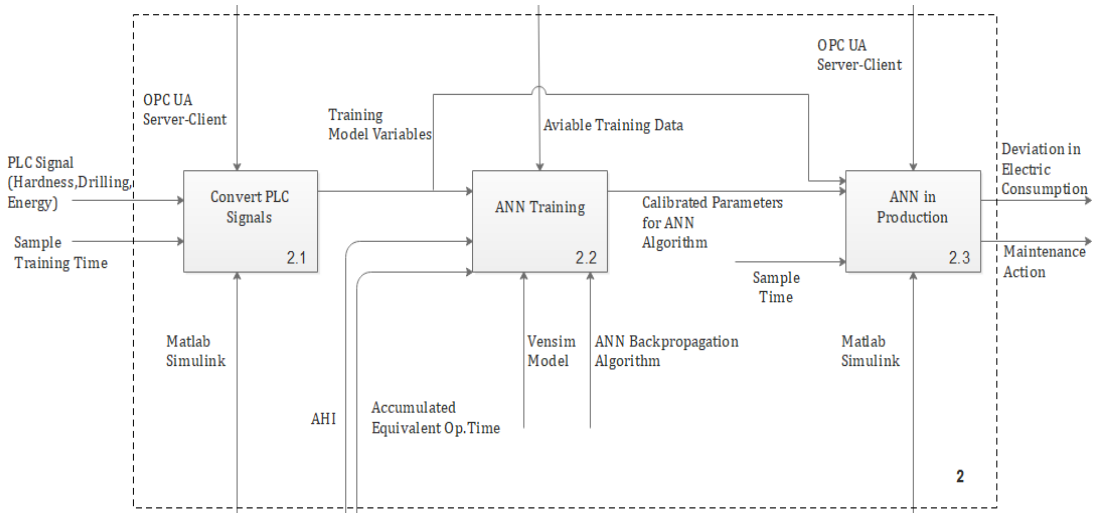


Figura e.6. Diagrama IDEF1 del proceso 2 del diagrama *IDEFØ*

Tras la exposición general del trabajo a grandes rasgos, se procede a mostrar en primer lugar el entorno en el que se ha desarrollado la tesis, y posteriormente el contenido de los modelos.

Desarrollo de los modelos del trabajo

Adquisición de Datos *(IDEF1 1.1)*

Para la adquisición de datos el laboratorio adopta OPC Unified Architecture como se ha indicado anteriormente, Matlab/Simulink sería por lo tanto un OPC UA cliente, con acceso a la OPC ToolboxTM que permite el acceso al histórico de datos.

Gracias a los PLC de las máquinas que funcionan como OPC UA servidor, seproduce la conexión a tiempo real de Matlab con el servidor que sería el PLC (Figura e.7). Para ello existe la función de Matlab/Simulink **opcua()** a partir de la cual se anidan sucesivas para acceder a la información deseada en un nodo específico. Junto a esto se utiliza Una función de MATLAB S nivel 2 que es la función que define las propiedades y comportamiento de una instancia de un bloque de Nivel-2 S-función de MATLAB que hace referencia a la función MATLAB en un modelo en Simulink MATLAB (Single Machine Model).

La misma función MATLAB compone de un conjunto de métodos de devolución de llamada que el motor de Simulink invoca al actualizar o simular el modelo. Los métodos de devolución de llamada realizan el trabajo real de inicializar y computación las salidas del bloque definido por la función.

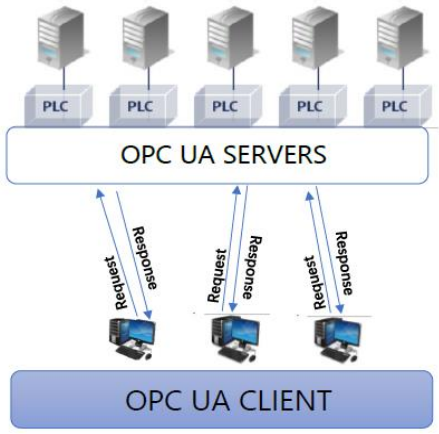


Figura e.7. Estructura OPC UA Cliente - Servidor

Single Machine Model/Machine state determination *(IDEF1 1.2)*

En este trabajo se ha realizado el modelado en base a una sola máquina, y la seleccionada es la Drilling Machine (Máquina taladradora). Este modelo se ha realizado en Matlab/Simulink utilizando la librería SimEvents, que se utiliza para la creación de modelos y bloques de simulación de eventos discretos. Para comenzar el modelado, el diseño del sistema en el Laboratorio, permite modelar cuatro estados:

- *Idle*: Se mueve la cinta transportadora, pero no se realiza ninguna operación.
- *Working*: La máquina realiza su operación y trabaja.
- *Energy-saving mode*: La energía consumida es la cantidad necesaria solo para mantenerla encendida.

- *Fault*: La máquina está bloqueada por un comportamiento anormal y muestra mensaje de error en la interfaz human-machine (HMI). Solo cuando se corrige este error la máquina abandona este estado.

Una vez definidos los estados, se debe realizar la conexión con el equipo en el Laboratorio. La conexión Matlab/Simulink con PLC/Sensor se realiza con OPC UA como se ha explicado anteriormente. La función 2 Level S-Function de Matlab se utiliza junto a las funciones de OPC UA para seleccionar los sensores necesarios dentro de los más de 100 existentes en cada estación, que sirvan para modelar el estado de manera eficiente. Finalmente con tres sensores tras muchas pruebas se consigue identificar los estados (XQA A1, xBG1, iRedCode). Los valores de estos a la salida de la función de Matlab es 1 cuando están en valor TRUE y 0 en FALSE. El primero de los sensores XQA A1 representa la cinta de la cadena de montaje, el segundo xBG1 indica cuando el MES le indica a la máquina realizar su trabajo; por último iRedCode toma el valor 4 cuando existe un error.

xQA A1	xBG1	iRedCode	State
1	0	0	IDLE
1	1	0	IDLE
0	1	0	WORKING
1	1	4	FAULT
1	0	4	FAULT
0	1	4	FAULT
Others			ENERGY SAVING

Tabla e.1. Relación de valores de entrada en sensores y estados de la máquina

Estos valores de los sensores son tomados por la Level 2 S-Function, y después llegan cómo inputs a una función de Matlab que genera un valor de salida dependiendo del estado en el que se encuentra la máquina (Tabla e.1). Para tener un mayor nivel de precisión se ha seleccionado una medida de tiempo (simple time) en este caso de 0.4 segundos.

SIGNAL → Matlab Function Variable (INPUT)

Conveyor == belt
xBG1 == xBG1
iRedCode == error

Matlab Function OUTPUT

Y==1 → IDLE
Y==2 → WORKING
Y==3 → ENERGY SAVING
Y==4 → ERROR

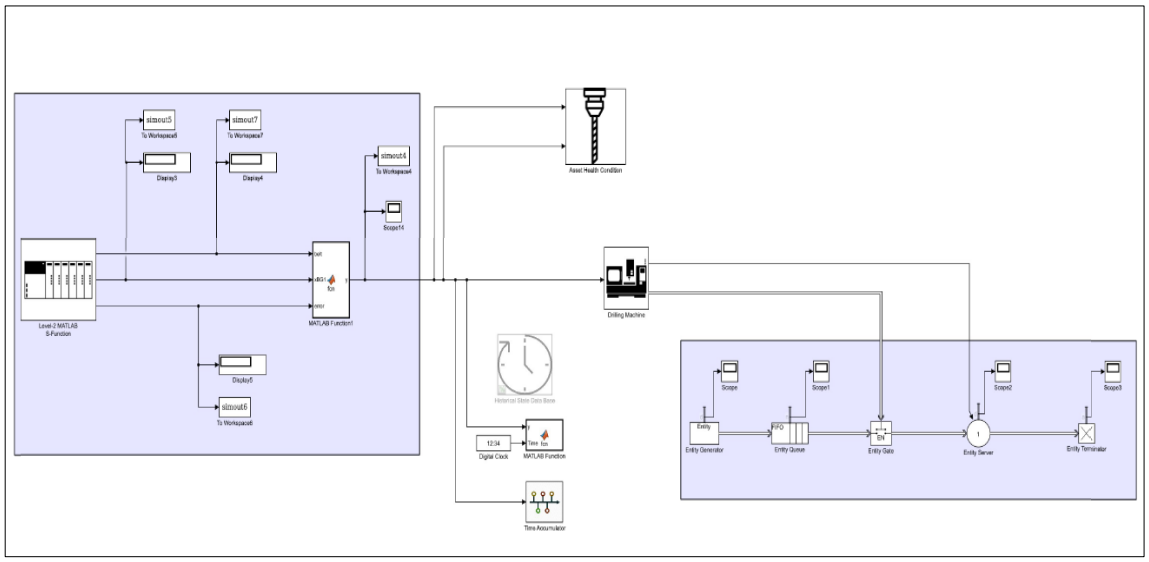


Figura e.8. Modelo Matlab Simulink para Drilling machine

Lo explicado anteriormente recoge el cuadro de la esquina superior izquierda del modelo, que representa la obtención de los datos del sistema físico para desarrollar el digital (Figura e.8). Posteriormente se ha realizado la programación de los dos bloques inferiores que se encuentran en la zona central. El primero de estos se ha realizado para monitorizar el estado en el que se encuentra la máquina a tiempo real. Recibe como input el tiempo real por medio del bloque "digital clock" y el estado de la máquina que lo recibe de la función de Matlab anterior. Esta función en cuestión tiene como salida la generación de un fichero txt que escribe a tiempo real el tiempo y el estado en dos columnas paralelas.

Con el fin de la función que genera el histórico de estados a tiempo real en el DT, se plantea la realización de un acumulador de tiempos en cada estado, pues hasta este punto el DT solo tiene una función de monitorizar a tiempo real. Para poder hacer un estudio profundo del rendimiento de la máquina, y para poder ser utilizado posteriormente como indicador en el modelo de salud del activo, se realiza el acumulador. La función del acumulador tiene los mismos inputs que la que monitoriza el histórico, con la diferencia de que esta función va generando un sumatorio del tiempo que pasa la máquina en cada estado. Finalmente se obtiene un txt en el cual en cada fila (simple time), se acumula el tiempo, y en la última se acumula por lo tanto el tiempo en cada estado en total en el tiempo que se haya realizado la simulación.

Con lo cual, al final de esta parte, el DT ya realiza una función de monitorizado y acumulador como si fuera un espejo de la realidad, y nos ofrece una información muy importante de cara al estudio del ciclo de vida de la máquina o activo. A partir de este punto el DT va a evolucionar a estudiar en consumo de la máquina y posteriormente a implementarse un modelo para el estudio de la salud del activo.

Drilling Machine Condition Modelling *(IDEF1 1.3)*

Como se menciona en el párrafo anterior, y en el esquema IDEF0, en esta sección se desarrolla el modelo gemelo de manera que este pueda ser una herramienta útil y relevante para el estudio de la salud del activo y la condición de este.

Por las características de las máquinas del laboratorio (que es meramente académico), la máquina de taladrado "Drilling Machine" no realiza un trabajo real, pues simula el taladrado (Figura e.9). Por esto, la máquina no se degrada pues no realiza realmente el trabajo en cuestión. En base a esto, y con el fin de desarrollar el modelo que se presenta a continuación, se ha simulado la existencia de dos factores que definen las características del trabajo y suponen una degradación en el activo. Estos dos factores son "Hardness factor of the material" (Factor de dureza del material) y "Drill factor of the material" (Factor que define el número de agujeros que realiza el talado de la Drilling Machine). El primero de estos simula la dureza que tiene el material sobre el que se realiza el trabajo, y pondera la degradación que supone esta dureza en tres niveles de valor dependiendo de la dureza del material. El segundo factor mencionado simplemente define en una escala del 1 al 4, el número de agujeros que realiza el taladro.

Figura e.9. Máquina de taladrado (Drilling Machine) del Laboratorio de Industria 4.0

Cuando se definen los factores de degradación, estos junto al tiempo real de simulación y al estado en el que se encuentra la máquina son los inputs que permiten generar el modelo. La capacidad de monitorizar del modelo DT muestra una gran importancia llegados a este punto, pues para el estudio de la salud del activo y su condición, el modelo considera que el activo o máquina en cuestión sólo degrada su condición cuando se encuentra en el estado de trabajo. Esto se considera así, pues el propio talado solo puede degradarse cuando realiza un trabajo de taladrado, y tanto en el estado Idle, como en Energy Saving, el taladro no realiza una operación que implique un desgaste.

El modelo de condición en cuestión, utiliza indicadores que van anidándose con el paso de profundidad de los mismos, para finalmente conseguir una variable o indicador que pueda monitorizar la condición de salud del activo. En primer lugar o nivel de indicador, se encuentran los dos factores simulados ("Hardness factor of the material" y "Drill factor of the material") así como un acumulador de tiempo en cada

uno de los estados que ya se muestran en el Matlab Simulink DT. En este primer nivel se puede apreciar una unión de los factores reales junto a los dos simulados, y la monitorización de estos.

En el segundo nivel, comienzan a anidarse los indicadores tal y como se menciona anteriormente. Está en primer lugar un indicador llamado "Equivalent operating time" (DH) que multiplica los valores del factor de dureza y el de taladrado, cuando el estado es de trabajo. Esto representa que la degradación del activo depende del trabajo que realiza la máquina y del tipo de trabajo en sí mismo que puede llegar a suponer un desgaste mayor que el estipulado en un determinado tiempo, dependiendo del estrés o intensidad que supone este en el activo en cuestión. A parte de este indicador que es el principal para el desarrollo del modelo, se incluyen en este segundo nivel otros relacionados con disponibilidad y utilización.

En el tercer nivel se presenta un indicador importante "Accumulated Operational Time" (AOT) que realiza la acumulación del tiempo operativo de trabajo. Este indicador acumula en sumatorio del valor de DH, y toma el valor nulo cuando aparece un fallo, y esta variable solo va a sumar tiempo a la acumulación cuando se encuentre el activo en estado de trabajo. Además de este indicador existe otro para hacer un porcentaje de utilización de la máquina en el tiempo real de simulación. Entre este nivel, y el último del modelo, se desarrolla y define el indicador que va a monitorizar la condición del activo (Asset Health Index).

En primer lugar se supone una tasa de deterioro que es única para cada activo y que viene determinada por la ecuación:

$$Tasa\ de\ envejecimiento(AR) = \frac{\ln \frac{H\ vida\ esperada}{H\ nuevo}}{Vida\ esperada}$$

Dónde:

- *H vida esperada* = 5,5 .
- *H nuevo* =0,5.
- *Vida esperada* es la estimada para ese activo antes de su overhaul o remplazo=10.

Este Ageing Rate (AR) o Tasa de envejecimiento, junto a el tiempo acumulado de operación (AOT), y la Hn (Horas nuevo que representan la condición inicial no ideal de la máquina o activo); sirven para modelar el Indicador de Salud del Activo (AHI).

$$AHI_t = Hn \cdot e^{AR \times AOT_t}$$

Hasta este punto del modelo, ya se puede realizar una monitorización de cómo cambia la condición del activo (Teniendo en consideración que los dos factores de degradación en este caso son simulados). Y una vez logrado esto, se finaliza el modelo con un nivel de indicadores que utiliza el indicador de salud AHI para estudiar la evolución de la probabilidad de fallo.

$$PoF_t : \begin{cases} 0.01 & AHI_t \leq 5{,}5 \\ 0.0000359381 \times (e^{1.02337 \times AHI_t}) & 5{,}5 < AHI_t < 10 \\ 1 & AHI_t \geq 10 \end{cases}$$

Por último el modelo presenta una doble visión a su fin. Por un lado se define una determinada política de fallo, que define un límite de Probabilidad de Fallo en el activo a partir del cual se realiza la parada para realizar preventivo (Con el objetivo cero-breakdown). Y por otro lado en un plano relacionado con la monitorización del fallo, se puede realizar una simulación Montecarlo con un número aleatorio y la variable Fail (fallo), lo que aludiría a un DT enfocado al prototipado o monitorización más que a la toma de decisiones, pues la toma de decisiones sestaría más presente en el primer enfoque.

Machine artificial Neural Network *(IDEF1 2.2)*

En primer lugar, se obvia en este resumen la explicación del proceso *IDEF1 2.1*, pues es análogo al explicado anteriormente. Cómo se ha explicado de la red neuronal, es una técnica de vigilancia, del proceso de aparición y propagación del fallo, que se basa en desviaciones observadas en el taladro, de lo que sería su modo de comportamiento normal para una determinada condición operativa. Por ejemplo sabemos que el consumo eléctrico del taladro, para una dureza de material y un número de taladros debería estar entorno a x (porque hemos entrenado una herramienta que nos lo sabe calcular), y sin embargo hemos anotado que está en x+incremento x, entonces podríamos concluir que el proceso de fallo se está desarrollando, y cuánto tiempo puede durar.

Se diseña por lo tanto una red neuronal que como salida tenga el consumo eléctrico, y como entrada tenga el número de taladros, el material, el tiempo operativo de la máquina, y el AHÍ; y entrenarla para predecir bien el consumo. El diseño de esta herramienta, y su coordinación con la de AHI, permite tener un mejor modelo de salud de activos calibrando mejor nuestros multiplicadores de efecto de taladros o dureza de material, etc. Con la red neuronal podemos detectar con antelación los fallos antes de que físicamente se produzcan.

Se utiliza el proceso de entrenamiento de backpropagation. El objetivo de este proceso es actualizar cada uno de los pesos y valores de la ANN, para que generen un output cada vez más cercano (menor error) al output ideal. Esto se logra minimizando la función del error para la neurona output.

El esquema está constituido por neuronas interconectadas y arregladas en tres capas. Los datos ingresan por medio de la “capa de entrada”, pasan a través de la “capa oculta” y salen por la “capa de salida”. Cabe mencionar que la capa oculta puede estar constituida por varias capas a su vez. La neurona artificial pretende imitar las características más importantes de las neuronas biológicas. Cada neurona i-ésima está caracterizada en cualquier instante por un valor numérico denominado valor o estado de activación; asociado a cada unidad, existe una función de salida, que transforma el estado actual de activación en una señal de salida. Dicha señal es enviada a través de los canales de comunicación unidireccionales a otras unidades de la red; en estos canales la señal se modifica de acuerdo con la sinapsis (el peso, w) asociada a cada uno de ellos según determinada regla.

Para implementar este mecanismo de entrenamiento se ha utilizado un modelo dinámico de simulación en el programa Vensim. Para ello se realiza la implementación del modelo de ANN en un modelo dinámico de simulación, utilizando la nomenclatura y estructuración propias de este.

Cómo se ha indicado anteriormente, debido al carácter simulado de los factores de trabajo y degradación de la máquina (En el lab 4.0 la máquina de taladrado solo simula el trabajo de perforación) la red neuronal y su modelo, no se han implantado en el DT pues los datos recogidos y de entrenamiento son simulados. Se ha realizado por lo tanto un planteamiento y ejemplificación de cómo el proceso debe realizarse hasta el punto de su implementación en un sistema gemelo virtual real de un sistema físico.

Resultados del trabajo

El resultado principal de este trabajo es la metodología que se ha mostrado representada tanto por procesos separados como integrado en un modelo digital, formando un DT con simulación multidisciplinaria; así como los propios resultados de cada uno de los modelos.

En primer lugar, para realizar la conexión del Matlab con el PLC, el primer resultado efectivo o relevante del trabajo, es la conversión de las señales del PLC a variables del modelo mediante la función Matlab Level 2 S-Function y OPC UA expert. Tras la consecución de esto, se consigue modelar de manera efectiva los estados de la máquina, y a su vez monitorizar a tiempo real el estado de la máquina en cada instante del tiempo y representar todo esto mediante un archivo txt. A parte de esto, el siguiente resultado relevante es la acumulación del tiempo en cada estado y su representación igualmente en txt. A parte de estos archivos txt que representan la monitorización, el bloque scope de Matlab Simulink permite ver en los bloques del modelo a evolución de sus valores durante la simulación, al final de esta.

Time	Statevalue
0.00	3.00000000
0.40	3.00000000
0.80	3.00000000
1.20	3.00000000
1.60	3.00000000
2.00	3.00000000
2.40	3.00000000
2.80	3.00000000
3.20	3.00000000
3.60	3.00000000
4.00	3.00000000
4.40	3.00000000
4.80	3.00000000
5.20	3.00000000
5.60	3.00000000
6.00	3.00000000
6.40	3.00000000

IdleTime	WorkingTime	EnergySavingTime	FailureTime
0.00000000	0.00000000	0.40000000	0.00000000
0.00000000	0.00000000	0.80000000	0.00000000
0.00000000	0.00000000	1.20000000	0.00000000
0.00000000	0.00000000	1.60000000	0.00000000
0.00000000	0.00000000	2.00000000	0.00000000
0.00000000	0.00000000	2.40000000	0.00000000
0.00000000	0.00000000	2.80000000	0.00000000
0.00000000	0.00000000	3.20000000	0.00000000
0.00000000	0.00000000	3.60000000	0.00000000
0.00000000	0.00000000	4.00000000	0.00000000
0.00000000	0.00000000	4.40000000	0.00000000
0.00000000	0.00000000	4.80000000	0.00000000
0.00000000	0.00000000	5.20000000	0.00000000
0.00000000	0.00000000	5.60000000	0.00000000
0.00000000	0.00000000	6.00000000	0.00000000
0.00000000	0.00000000	6.40000000	0.00000000

Figura e.10. Histórico y acumulador de tiempo

Además de la información sobre los estados de la máquina, este modelo también provee el valor del consumo energético de la máquina (Figura e.10).

Hasta este punto el DT tiene una función de monitorización, y a partir de aquí con el Drlling Machine Condition Model, se comienza el estudio y los resultados de la condición de salud del activo. En primer lugar, debe aclararse que para la implementación del modelo se ha simulado los factores mecánicos de operación (Hardness, Drill), y que debido al carácter simulado, la orden de parada de mantenimiento no se ha creado como señal que genera una acción de vuelta en el PLC, y no se ha definido el límite en el modelo como política de permisividad de fallo. Con lo cual este modelo, a falta de ciertos datos reales que el Laboratorio Industria 4.0 no ofrece, calcula el valor del AHI, la probabilidad de fallo, el tiempo equivalente operativo acumulado; y a su vez el valor de ambos factores simulados. Estos outputs se representan de la misma manera que los presentados anteriormente mediante ficheros txt, y entidades del modelo.

En términos de la ANN, se han realizado simulaciones en Vensim con datos independientes de este trabajo, para comprobar la validez del entrenamiento de la red con diferentes números de neuronas. De esta manera, se calibra el valor de los parámetros del modelo de la red neuronal, y se comprueba el funcionamiento de esta con el resto de datos existentes. Debido a la ausencia de datos reales de los factores de operación, esta ANN no se ha implementado en el modelo DT de Matlab, y solo se ha planteado cómo sería su estructura, inputs, y output.

Conclusiones y futuros trabajos

Este trabajo ha servido para llegar a varias conclusiones, y a su vez plantear un trabajo y desafío futuro de desarrollo de las herramientas planteadas.

En primer lugar se llega a la conclusión de que existen tres planteamientos principales para la utilidad y uso del DT. El primero es la utilización del DT como un mero espejo virtual que es capaz de monitorizar todas las variables modeladas en torno al funcionamiento de la máquina. Tras esto podríamos definir una segunda función del DT como herramienta para la toma de decisiones de mantenimiento basadas en la condición del activo y su salud. Y finalmente existe un enfoque del DT como herramienta de prototipado virtual, que permite el desarrollo de prototipos virtuales que hacen la misma función que un sistema físico y permite el ahorro de dinero y mayor eficiencia a la hora de desarrollar nuevos productos.

Tras esta definición, se puede observar que en este trabajo se ha logrado el desarrollo del DT como herramienta de monitorización del comportamiento de la máquina, a la vez que la condición del activo. Pero por falta de capacidades del activo en cuestión y datos reales, no se ha implementado en el modelo la herramienta que se utilice para realizar el envío de la señal de vuelta al sistema físico desde el DT para la ejecución de acciones de mantenimiento.

Por último este trabajo propone el reto y futuro desarrollo de nuevas herramientas de vigilancia y toma de decisiones para el mantenimiento y la gestión de activos, mediante la aplicación de redes neuronales. Se puede apreciar la utilidad de esta herramienta y la factibilidad que tiene si se tienen los datos necesarios para ello. Y este ámbito de digitalización combinado con herramientas como la ANN propias del ámbito del Machine Learning, inspira un enfoque futuro a un mantenimiento inteligente en el cual exista una fluida conexión entre el mundo físico y el virtual que

permita que la máquina se auto entrene con sus propios datos, y pueda generar un mantenimiento mucho más eficiente en todos los aspectos y con zero-breakdown.

CHAPTER 1. INTRODUCTION AND OBJECTIVES

1.1. Introduction

In the most recent years, digital technologies have developed in such a high pace that push and challenge the traditional manufacturing industry to adapt to and adopt them in order to stay competitive in the market. In these terms, the manufacturing world is experiencing a new paradigm shift, known as the fourth industrial revolution: “Industry 4.0”.

This new wave is based on the widespread adoption of information and communication technologies such as Internet of Thing (IoT), Big Data Analytics, Smart Sensors. Together with computer simulation, it will make the manufacturing environment extremely networked and interconnected, and able to make decentralized and automatic decisions with the guarantee of completeness of information and data reliability. To realize the data continuity through the production lifecycle, Cyber Physical System (CPS) has been deployed more and more as a strategic move by the company. It is a system which consists of a physical part associated with a cyber one, embedded with storage, computational capability. In CPS, information from all related perspectives along the lifecycle are closely monitored and continuously synchronized in real-time between the physical factory floor and the cyber computational space: the “Digital Twin (DT)”, the digital representation of the physical world inside the factory, it updates parallel to the physical one. With the use of these technologies, Industry 4.0 opens the way to real-time monitoring and synchronization of the real world activities to the virtual space thanks to the physical-virtual connection and the networking of CPS elements.

Simulation modelling is the method of using models of a real or imagined system, to better understand or predict the behaviour of the modelled system or process. Today, the use of simulation modelling in science and engineering is well established, especially in manufacturing field. They are adopted widely in recent years thanks to its ability to solve the stochasticity in the manufacturing system and to predict and improve the business system behaviours. Discrete event simulation (DES) is one of the most commonly used simulation techniques for analysing and understanding the dynamics of manufacturing systems. However, with the increased integration of simulation modelling in the product life cycle management, the user requirements have changed considerably. Increasing product variants and customisable products require more flexible

production systems. The advent of the Industry 4.0 has brought changes to the simulation modelling paradigm as well.

The DT means the virtual and computerized counterpart of a physical system that can be used to simulate for various purposes in different scenarios, exploiting a real-time synchronization of the sensed data coming from the field.

With the creation of the DT, companies may realize significant value in different areas, such as reduce leadtime, improved quality etc. They may realize value and benefits iteratively and faster than ever before. But, before anything else, enterprises need to understand the definition and the approach to develop the DT in order to avoid being overwhelmed. Unfortunately, scientific literature that describes the contextualisation of the concept in the manufacturing domain is still at its infancy, there is still not a unique and shared definition of DT in both research and industrial field, let alone the standard method to implement the DT. For this reason, this work proposes to deal with a research activity about the clarification of the DT concept and to provide a step forward in the implementation of such DT model in a manufacturing context.

1.2. Thesis Motivation

Nowadays the world is being witness of an all level digitalization that guides the industry and business to a restructuration, in order to adaptate to the new requirements of the surrounding enviroment. That change also concerns to the labour of the technical professionals, and their formation. As a consecuence of this deep consciousness-raising, this thesis tries to investigate and develop simulation models based on the current digitalization.

The process of turning a factory into a digital factory is also fostered by the client´s requirements that are basically less waiting time and more customization. And obviously, with a deeper control of the factory and the production at real time, we take the path to reach efficiency within a company, diagnosis and maintenance time minimization, and delays due to unavailability reduced.

Motivated by all the reasons above, the aim of these thesis is the developement of “Digital Twin” models providing real time results, that could turn into basis for the industrial management decisions; and place them in the Industry 4.0 paradigm enviroment, as well as in the european research project MAYA, considered as one of the many research works aiming to promote the new simulation paradigm called “Digital Twin”.

1.3. Thesis Background

This work is grounded on several key aspects:

- The analysis of the state of the art of simulation technologies used in the manufacturing sector;
- The DT initiative in the industry 4.0 context. DT is still at the infancy stage, even if the vision of the DT has evolved during the last decades, there is not a unique definition/interpretation of the DT concept both in research and industry, which results in no-common methodology or procedures to implement it;
- The simulation aspect of the DT, which can be seen as the future technology in the field of simulation and plant status forecasting.
- The explotation of real time data, gathered from the shop floor, into the simulation environment could lead to great advantages in decisions-making. With regards to the DT, a real-time synchronization with the real system has not been achieved yet. Neither is the digital representation used to understand, in real-time, what is happening on the shop floor, nor is the real system updated with improvements that may have been achieved in the digital model.
- The general objectives of the MAYA project, that will be later explained, and more specifically with the definition provided by some developments of MAYA project, trying to cover the gaps between the current state of the art and the future of simulation and forecasting technologies.

1.4. Thesis Objetives

This work proposes an innovative interpretation of the DT concept and the development of a possible DT simulation model that together with the reference framework, represents the key enablers to implement real multi-disciplinary simulations to improve manufacturing aplications.

In the first place and to understand the evolution of the DT concept, this work's initial objective is to have a literature analysis on the topic and on similar concepts.

With respect to the DT applied to simulation, this work benefits of the possibility to stay during four months at the "Industry 4.0" Lab in Politecnico di Milano, offering me the chance to develop a simulation model which grabs production line real-time data and processes it, in parallel computing, for a better understanding of real-time equipment condition, reliability and risk. Thus, this work takes a step forward compared to the traditional simulation

models by adopting the real-time data instead of history data or statistical distributions. The idea is to build up the basis for creating a DT simulation model. Physical-virtual synchronization allows updating digital factory simulation which attempts to predict how the system should work into a digital factory replication, which shows how the system is actually working. And thank to the models built on Matlab´s Simulink with the SimEvents library, and the Industry 4.0 Lab in Politecnico di Milano a field validation of the DT model has been made possible. At the same time, this work's objective is also to develope a simple simulation model that allows to study the further aplications of the DT to the manufacturing and smart maintenance area.

1.5. Summary

The thesis is structured as follows:

In these first Sections a brief description of this project's environment is offered; this easy the interpretation for the entire work, sets its purpose, the methods for tools representation and usage, and explains the meaning of numerous terms of which extensive use in the remaining work's stages has been made.

In chapter 2 the state of the art is presented, with all the concepts exploited to reach the main purpose of the thesis, in industry 4.0 area. Focused in the midst of a fourth wave of technological advancement, the rise of the integration of digital industrial technologies such as Internet of Things (IoT), and Big Data and Cloud Computing that leads to a smart and interconnected manufacturing paradigm shift. The legacy systems will be transformed to cyber physical system (CPS), a system which is intelligent, self-aware and able to support decision making.

In chapter 3 all the acknowledgments exploited to reach the main purpose of the thesis are reported: a review of the literature contributions is offered and, in the detail, tools and methods that represent the basis of the work are briefly explained and contextualized to have a deeper understanding of the Digital Twin concept and the cyber physical systems. However, the scientific literature does not provide a unique and shared definition of this concept, let alone a standard method to implement this tool.

Chapter 4 includes the different phases of the work development, starting from an aggregated level of detail, main decision phases have been acknowledged and reported through the IDEF0 representation methodology. Following, work basis and structure is justified providing an explication of the meaning and the different dimensions of the thesis models main objective.

Afterwards **chapter 5**, the field validation of the work is explained, by presenting the Politecnico´s Industry 4.0 Lab and all the supporting tools from digitalization that it provides, to develop the DT. In this way, the next to be explained is the DT model in Matlab Simulink, going deep into the model developing and the components from it. To en with the model, some of the functions and facilities that it provides are shown.

Afterwards the thesis presents the development of the "Machine Condition Model", which represents the health of the asset through the time by the Asset Health Index (AHI), and the probability of failure of the asset when the accumulated operation time increases. Thus the model mentioned before is implemented as part of the DT model in Matlab Simulink. With this model the tesis will reach to monitor the health of the asset by the AHI.

Apart from that, we can also use other vigilance techniques focused on the process of appearance and propagation of the failure, based on deviations from the normal behaviour on a specific condition observed on the asset. In order to develop the idea, a solution made by an Artificial Neuronal Network model is suggested.

In **chapter 6** the results are presented and assessed arguing the contribution in different ways and providing the meaning and the value of the efforts.

In **chapter 7** a conclusion and a brief summary are offered to enclose the job in a few simple concepts and present possible future development of the efforts accomplished.

CHAPTER 2. TECHNOLOGICAL BACKGROUND

2.1. Industry 4.0

2.1.1. Introduction to the concept

Ever since the beginning of industrialization, technological leaps have led to paradigm shifts which are addressed today as "industrial revolutions", in the field of mechanization with the introduction of steam power in 18th century (so-called the 1st industrial revolution), of the intensive use of electrical energy and mass production in the final third of the 19th century and the beginning of the 20th (so-called the 2nd industrial revolution), and of the digitalization and automation at the end of 20th century (so-called the 3rd industrial revolution). Today, on the basis of an advanced digitalization within factories, the integration of internet technologies and future-oriented technologies seems to push a new fundamental paradigm shift in industrial production. Therefore, manufacturing world is experiencing the fourth industrial revolution, this revolution will profoundly change the dynamics of the industrial sector [1][2].

Industry 4.0 (so called the fourth industrial revolution) was stated for the first time during the Hannover Fair in 2011 [3]. It originates from a project within the German government's high-tech strategy, which promotes the computerization of manufacturing. However, since its first announcement, Industry 4.0 has been one of the most discussed topics in both research and industry. This conceptual idea has brought new initiatives world-widely and been adopted also by other industrial nations within the European Union, for example, "Industria Conectada" in Spain, "Smart Industry" in Netherlands, and further "Chinese manufacturing 2025" in China, India, and other Asian countries. Even if it may be called in a different way, they all refer to an innovation in the industry sector pushed by the emerging technology breakthroughs: such as Internet of Things (IoT), Cloud Computing, Big Data, Robotics etc [4]. The basic principle of Industry 4.0 lies on the core of IoT and smart manufacturing: "work in progress products, components and production machines will collect and share data in real time".

2.1.2. Pillars

As anticipated before, the major technical background of Industry 4.0 is the introduction of internet technologies into industries: Internet of Things, Big data, Cloud computing. In combination with the simulation technologies, they form the building blocks of Industry 4.0. Many of these technologies

have already been used in manufacturing, but with Industry 4.0, they will be connect seamlessly and transform production. Currently isolated, optimized cells will come together as a fully integrated, automated and optimized production flow, leading to greater efficiencies and changing the traditional production relationships among suppliers, producers, and customers as well as between human and machines [5].

Big data is a loosely defined term to describe data sets that are so large and complex which are difficult to be analysed using standard statistical software. This big amount of data can be used to optimize production quality, save energy, and improve equipment service. In Industry 4.0 context, the collection and comprehensive evaluation of data from many different sources will become standard and key information to support real-time decision making.

Internet of Things (IoT), also known as the Industrial Internet of Things (IIoT) in manufacturing domain, is an "emerging global Internet-based information architecture facilitating the exchange of goods and services, with the function to overcome the gap between objects in the physical world and their representation in information systems"[6]. This technology allows manufacturer's sensors and machines being networked and making use of embedded computing. In Industry 4.0 context, more devices, including even unfinished products, will be enriched with embedded computing capacity and connected using standard technologies. This allows field devices to communicate and interact both with each other and with more centralized controllers, as necessary. It also decentralizes analytics and decision making, enabling real-time responses.

Cloud Computing is defined by the NIST (National Institute of Standards and Technology) as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction". It builds the bridge of data sharing across sites and companies, even across countries. With Industry 4.0, machine data and functionality will increasingly be deployed to the cloud, enabling more data-driven services for production systems. Even systems that monitor and control processes may become cloud based [5].

2.1.3. Industry 4.0 on simulation

In manufacturing context, simulation modelling is defined as "the imitation of the operation of a real-world process or system over time. Simulation involves the generation of an artificial history of the system, and the observation of that artificial history to draw inferences concerning the operating characteristics of the real system that is represented"[7]. In the past few decades, it has already been widely used in the engineering phase: such as 3-D simulations of products, materials and production processes. It allows for the experimentation and validation of product, process and system design and configuration. However, increasing product variety and customizable products require more flexible production systems modelling. The advent of the Industry 4.0 paradigm has brought changes to the simulation modelling paradigm, simulations will be used more extensively in plant operations as well. These simulations will leverage real-time data to mirror the physical world in a virtual model, which can include machines, products, and humans, allowing operators to test and optimize the machine settings.

The new simulation modelling paradigm triggered by Industry 4.0 is best embodied by the "Digital Twin (DT)" concept, which represents the virtual and computerized counterpart of a physical system that can be used for various simulation purposes, exploiting real-time sensed data coming from the field. Every prospective of physical element, starting from a single machine up to a whole production facility level, will be accompanied by its virtual representations, which allow predicting its performance in the physical world. The digital counterpart will represent at any time its current condition and information created along its own value chain and will dynamically grow during its lifetime, made possible by the increasing digitalization in every stage of manufacturing. The DT concept extends the use of simulation modelling to all phases of the product life cycle. It contains all the information that is needed by various stakeholders and aligns it in a structured way, which means that all elements are semantically described and linked with relevant meta-information [8].

2.2. The Framework. The MAYA Project in PoliMI

European research project MAYA[1] (MultidisciplinArY integrated simulAtion and forecasting tools, empowered by digital continuity and continuous real world synchronization, towards reduced time to production and optimization) is one of the many researches that aim to promote this new simulation paradigm – Digital Twin. This research work is developed under this project. The project was born within the HORIZON 2020 European Framework Programme for Research and Innovation in the trend of Industry 4.0. The participants of this project are listed in Table 1. There are eleven members which include research center, Small and Medium-sized Enterprises (SME) and industrial companies. Four of them are from Italy, the others are from other European countries: Germany, Finland, Romania and so on.

N	Participant organisation name	Short	Type	Country
1	Technology Transfer System s.r.l.	TTS	SME	IT
2	SIEMENS Industry Software LTD	SIEMENS PLM	IND	IL
3	SIEMENS AG Corporate Technology	SIEMENS CT	IND	DE
4	Synesis S.C. a r.l.	SYNESIS	SME	IT
5	University of Applied Science of Southern Switzerland	SUPSI	RTD	CH
6	Politecnico di Milano	PoliMI	RTD	IT
7	Finn-Power OY	FinnPower	IND	FI
8	Deutsches Forschungszentrum für Künstliche Intelligenz	DFKI	RTD	DE
9	Volkswagen AG	Volkswagen	RTD	DE
10	Ropardo srl	Ropardo	SME	RO
11	Consiglio Nazionale delle Ricerche	ITIA-CNR	RTD	IT

Table 1: List of partners of the MAYA Project

MAYA addresses three high level objectives:

- MAYA for Digital Continuity;
- MAYA for the Synchronization of the Digital and Real Factory;
- MAYA for Multidisciplinary integrated simulation and modelling.

MAYA's first objective is to guarantee that relevant digital information from the field is preserved all along the factory life-cycle despite changes that the equipment, methods and tools may undergo, allowing data to be enriched and used when needed for each phase, from the integrated design of the product production system, through the production operation, till the dismissal/reconfiguration phase. By creating a full life-cycle data loop can bring economical, operational and knowledge advantages.

The second objective of MAYA project is to empower the synchronization of the digital world, made of template or simulation models, with the real factory, that is continuously changing. DT is able to adjust itself in real-time to perfectly mirror the situation in the real world thanks to the data gathered through all the production lifecycle, thus bring great advantages in terms of monitoring, optimization, efficiency and failure anticipation. And this work aims at giving a contribution regarding this objective to realize physical and virtual synchronization.

The third objective is to create an environment in which other simulation tools are integrated, different simulation tools will run and interact, establishing agreements on them as for their initialization, synchronization, events dispatching and termination. Therefore, simulation models will be interconnected and synchronized among each other and enable the exchange of relevant data with other tools at runtime.

2.2.1. Cyber-Phisical System

To better understand how DT works in the Industry 4.0 paradigm, it is necessary to address the Cyber-Physical System (CPS) concept (Figure 2). A Cyber-Physical System is an automated system that enables connection of the operations of the physical reality with computing and communication infrastructures. CPS for manufacturing facilitates optimization of product development and total control of production system through real-time exchange of all information required for production enabled by Internet of Things (IoT).

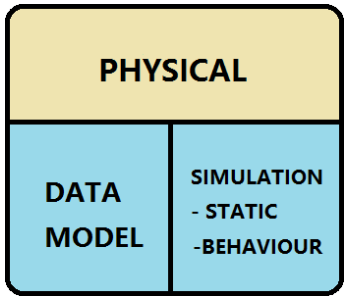


Figure 2: CPS System representation

Physical part: It is the physical part in the real world, which performs some tasks on workpieces in the real environment.

Cyber part: It is the virtual part of the CPS which can be seen as the DT. It consists of a data model which includes and structures all the information of the CPS through the production lifecycle, allowing data to be enriched and used as needed for each phase. Both static (geometry, other physical characteristics) and behaviour information are included. The second half of cyber part is the simulation models which can deal with different kinds of analysis. It contains not only static models such as digital assembly, material movements, they represent the core characteristics/basic functions of the system, but also the behaviour models such as FEM(Finite element method), DES (discrete event simulation), kinematics, energy and others. Data analytics are embedded in the cyber level with which CPS can take decentralized decisions through autonomously exchange information and computation and trigger action. Within Industry 4.0, the future production system will be CPS-based.

Communication network: Communication network includes the sensor network, the ubiquitous communication network and so on. It has the functions of data acquisition, transmission and communication in CPS. In particular, it allows CPS to communicate and exchange data with the Data Model.

2.2.2. Synchronization of the digital and real Factory

The project fosters the convergence of physical world and virtual world, where the latter must closely mirror the first and where the former generates an unprecedented volume of data to be handled by the digital representation of the Factory (Figure 3).

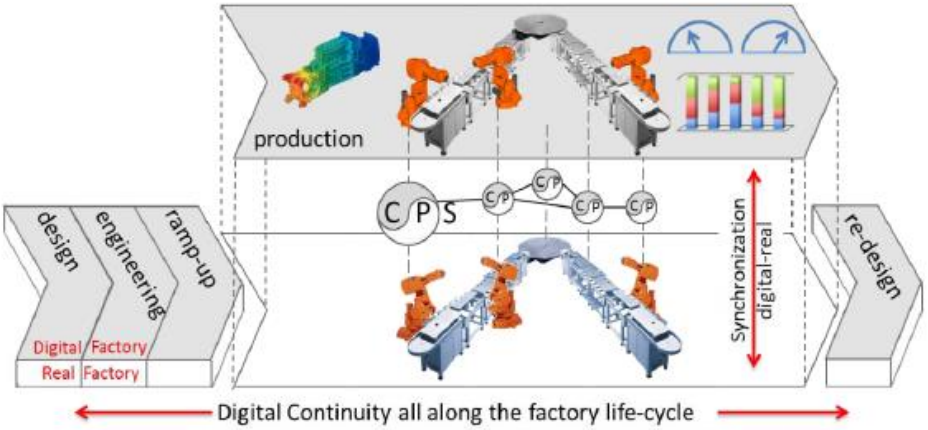


Figure 3: MAYA for synchronization of digital and real factory

MAYA will (1) propose a hardware infrastructure of intelligent middleware to extend legacy systems into the digital world; (2) define application protocol interfaces for the implementation of cross-tool secured communication drivers capable to provide both raw data streaming and high-level aggregation functions and (3) develop the software architecture to process large data sets.

These three sub-objectives will be framed within the context of the IEC-61499 international standard for distributed automation, by developing a HW/SW "CPS-izer", based on the technologies of the standard, specifically conceived to upgrade existing legacy systems (based on classical IEC-61131 technologies), either as an after-market add-in or directly integrated in normal PLC, to become effectively part of network of CPS connected to their digital twins.

2.2.3. **Current simulation practice in manufacturing**

Looking at the state of art of simulation technologies adopted today in manufacturing, discrete event simulation (DES) is one of the most commonly used techniques for analysing and understanding the dynamics of manufacturing systems. This type of simulation allows to assess the system's performance by statistically and probabilistically reproducing the interactions of all its components during a pre-defined time period and to support decision making, because it supports stochasticity modelling [9].

Applications of simulation in manufacturing system operation generally involve making shorter-term decisions when compared to the system design applications. In this phase, it deals with mainly operations planning and scheduling and real-time control [10]:

- *Operations planning and scheduling*. Operations planning and scheduling has received a tremendous amount of attention in the literature in

recent years. Operations planning and scheduling systems concern with the volume and timing of outputs, the utilization of operations capacity at desired levels for competitive effectiveness. DES simulation can be an effective tool to integrate loading, routing and dispatching issues to find the desired performance indicator: such as throughput, system utilization and mean flow time. Like in [11], a hybrid discrete event simulation and system dynamics (SD) were proposed by Jamalnia and Feili to model and simulate the aggregate production planning problem (APP), where the DES was used to model operational and shop-floor activities that are incorporated into APP and the estimated values of crucial time-based control parameters are the outputs used in SD.

While in most of the operations planning and scheduling studies, the literature assumed that machines were permanently available and reliable, in real life, machines can be unavailable for many reasons, including random breakdowns or scheduled maintenance operations, such disruptions during operations significantly affect the system's utilization and productivity. Therefore availability of the machine need to be considered during scheduling [12].

- *Real Time Control*: Real-time control of manufacturing systems is a difficult problem due to the complexity and stochastic nature of these systems. Simulation has proved to be an effective tool for real-time system control since simulation model can be integrated into the enterprise information system and the communication network. It can be also combined with optimization tools for decision making and evaluation. However, the use of simulation as a basis for a real-time system controller is still challenging due to various reasons: such as response time, data collection and aggregation issues make it an emerging field of research within manufacturing systems. Metan et al. developed a new scheduling system for selecting dispatching rules in real time integrating simulation, data mining, and statistical process control charts [13]. The proposed system was implemented on a job shop problem, with the objective of minimizing average tardiness, to evaluate its performance. The results showed that the performance of the new system was considerably better than the traditional simulation-based single -pass or multi- pass scheduling algorithm in terms of minimizing average tardiness.

- *Energy consumption.* In the manufacturing sector, energy consumption as one of the value-adding parameters in production has been given more and more attention. The principle drivers of this are: constantly increasing energy prices, the relate environmental impact such as CO2 emission and

new regulation policies. Therefore, in order to remain competitive in the market, companies need to adapt to these constraints and reduce production cost by increasing the energy consumption efficiency. But before optimizing it, there is a need to estimate it. Simulation can be a good solution in achieving this objective since it can build virtual production systems which emulate the actual system in real time, perform different experiments on different scheduling and production conditions and observe different effects [14]. As depicted inFigure 2.1, behaviour simulation should be also integrated in the DT simulation paradigm in order to monitor the system performance. However, although DES of manufacturing systems integrated with the relevant energy flows was identified as a promising approach, Herrmann et al. [15] and Thiede et al. [16] reviewed the use of commercially available DES tools for modelling energy consumption in manufacturing facilities. Both identified that currently available DES software do not support energy flow simulation as a standard. But several studies on approaches to integrate the energy flow into the material flow in DES simulation have been found in industrial production, they are divided in two directions: product oriented and production-state oriented.

A product-based simulation approach to estimate the energy consumption has been identified. Yingying Seow, et al, proposed a ESM (energy simulation model) to support energy consumption modelling within manufacturing system.

Another approach to simulate energy flow is operation-based, which means based on the states of the production. In [17], Weinert proposed the EnergyBlocks methodology aiming at integrating energy-efficiency criteria with evaluation and decision-making processes during production system planning and scheduling. The methodology was operation-states based, these states constitute the EnergyBlocks. Each state associates with its specific energy consumption. The representation of the production is a sequence of EnergyBlocks in sequence, and the energy consumption can be calculated as the sum of each block. In their work, only electrical energy was considered. In [18], authors designed a new production-state based approach to integrate material flow and energy consumption in DES software. In their model, not only electrical energy, but also the periphery systems like TBS (technical building services) were considered in the simulation as energy requirements from the asset. The model was validated in an automated line for the filling of sacks with powders by comparing simulation results on different parts sequencing.

2.2.4. Multidisciplinary integrated simulation and modelling

The project MAYA empowers the effective virtual validation of manufacturing equipment and systems prior to actual manufacturing, thanks to integration of models and simulation results from different domains as represented in Figure 4.

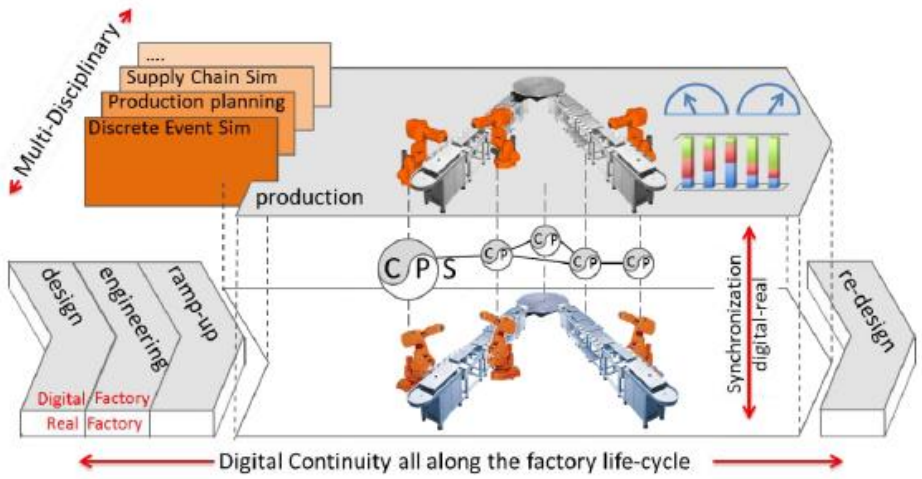


Figure 4: MAYA for multidisciplinary integrated simulation

The new envisioned meta-model, together with the reference framework, represents the key enablers to implement real multi-disciplinary simulations. MAYA will design and develop the software infrastructure for integrative interplay of models, so that each functional simulation model of the CPS will be able to publish and share relevant data, in a simulation as a service approach, feeding higher-level models and taking benefit from the results of lower-level ones.

2.3. The Digital Twin Concept

2.3.1. Introduction

By comparing the DT simulation triggered by the Industry 4.0 paradigm and the current practices of simulation technologies in the manufacturing sector, one can identify that classical simulation is characterized with stochasticity in nature and usually use history data and probability distribution to predict the parameters to be used in the simulation, then by carrying out multiple runs to evaluate the average performance so that to assess different scenarios or alternatives. On the other hand, DT simulation take a step forward by using the actual real-time data gathered from the shop - floor, for example, the machine reliability and availability is no more represented by percentage obtained from vendor and maintenance experience but will be acquired directly from the real system. In this way, it is possible to monitor simultaneously what is happening in the real manufacturing system and realize real-time synchronization of the system.

On the base of the classical DES of a manufacturing system, this work aims at giving a contribution as a first step to bring the DT concept into the simulation, in which machine states are synchronized with the real-time data coming from the shop-floor by smart sensors instead of percentage representation for simulative purposes. In this case, the simulation model is no more embedded inside the software system and controlled by a plant agent but it communicates directly with the real system thanks to the CPS initiative.

2.3.2. Literature Review

In this chapter, it has been done a literature review, to have a deeper understand of the Digital Twin concept. In this literature review, the following questions have been identified.

Main points:

- IP1 - When was the concept of “Digital Twin” brought up?
- IP2 - What is “Digital Twin”?
- IP3 - Are there any synonyms of “Digital Twin”?
- IP4 - What are the key enabling technologies of creating “Digital Twin”?

Regarding IP1, it concerns the starting date of the research. The term DT is one of the main concepts relates to Industry 4.0 wave which has been brought up in 2011 and it draws great attention in both the research and industrial world recently. Thus, this review aims to firstly figure out when did the new perception of DT was established. In this regard, the searching timespan has been limited from 2010 up to 2017. With respect to IP2, it is considered as the core of this review, to figure out what this concept represents and what can be benefited from this concept. Regarding IP3, since DToli is not a formal and shared concept in the research world, possible synonyms could be identified and some of them were discussed for the aim of comparison. With respect to IP4, the focus was laid on the manufacturing sector, papers belonging to other non-related sectors have been eliminated from the analysis and the main enabling technologies were extracted and summarized. Once the potentially relevant primary sources have been identified, they need to be assessed for their actual relevance, by going through all the titles and abstracts to extract the articles related only to manufactured products and production manufacturing system. In these papers, there were many which deal with aerospace industry, and in particular they focus on the DT for the material structure, which are not the focus here, so they have been excluded.

In this section, answers to the predefined points are discussed.

2.3.3. When was the concept of "Digital Twin" brought up?

The concept of DT is rather old. In 2003, the concept of a virtual, digital equivalent to a physical product or the DT concept was introduced during PLM (Product Life Management) course by Professor Michael Grieves at University of Michigan [19]. At that time, a digital representation was relatively new and immature. Not many studies have been conducted related to this concept.

It was then firstly used during the Apollo program, where two identical space vehicles were built. The first one was sent to the space, while the other one, the twin, was built and used on the Earth for training and for simulation before and during the space mission. As it is evident, the main scope of the original definition of the DT was to assist the astronauts in critical situations and to mirror the real space vehicles by considering stochasticity, historical data and sensor data, including interactions of the vehicle with the real world. In September, 2010, the concept of DT was addressed by NASA's integrated technology roadmap under the technology area 11: Modeling, Simulation, Information Technology & Processing, it states DT as "an integrated multi-physics, multi-scale, probabilistic simulation of a vehicle or system that uses the best available physical models, sensor updates, fleet history, etc., to mirror the life of its flying twin. It is ultra-realistic and may consider one or more important and interdependent vehicle systems." From that moment on, DT has been progressively used in mostly aircraft field but also more and more in other areas. Especially in 2013, the research of DT for the first time in manufacturing sector appeared. Jay Lee and his colleagues designed a DT model that operates in the cloud platform in the context of predictive manufacturing to simulate the health condition with an integrated knowledge from both data driven analytical algorithms as well as other available physical knowledge.

2.3.4. What is "Digital Twin"?

In spite of its simple formulation, the term DT opens up different interpretations between researchers because there is not a formal and shared definition in the research world. To have a complete view of the definitions of DT in the reviewed papers some different definitions of the terms are given:

- A digital representation of a real-world object with focus on the object itself [20].

- Ultra-high fidelity simulations to better understand both constraints in system design and possible consequences of external influences during the system's operation [21].
- A virtual representation of the real product, it has product's information since the beginning of the life until the disposal of the product [22].
- A Digital Twin is an integrated multi-physics, multiscale simulation of a vehicle or system that uses the best available physical models, sensor updates, fleet history, etc., to mirror the life of its corresponding flying twin [23].
- Model that operates in the cloud platform and simulates the health condition with an integrated knowledge from both data driven analytical algorithms as well as other available physical knowledge [24].
- A comprehensive physical and functional description of a component, product or system, which includes more or less all the information which could be useful in all the current and subsequent lifecycle phases [25].
- A living model of the physical asset or system, which will continually adapt to changes in the environment or operations and deliver the best business outcome [26].
- Digital representation of the physical environment [27].

From all these definitions, although they have different interpretations, it is clear that DT in general refers to a digital copy of a real object, resource or system. With the advent of the CPS, the DT appears as a virtual representation of the physical product which has all its information and knowledge, and connected with the physical part, allowing data transfer from the physical layer to cyber layer along the production lifecycle. Virtual simulated machine could receive data from the physical machine and send feedback after data processing to the physical machine during the manufacturing process [28].

From a future simulation point of view, DT concept can be seen as one of the next big advances in modelling, simulation and optimization technology, as illustrated [23].

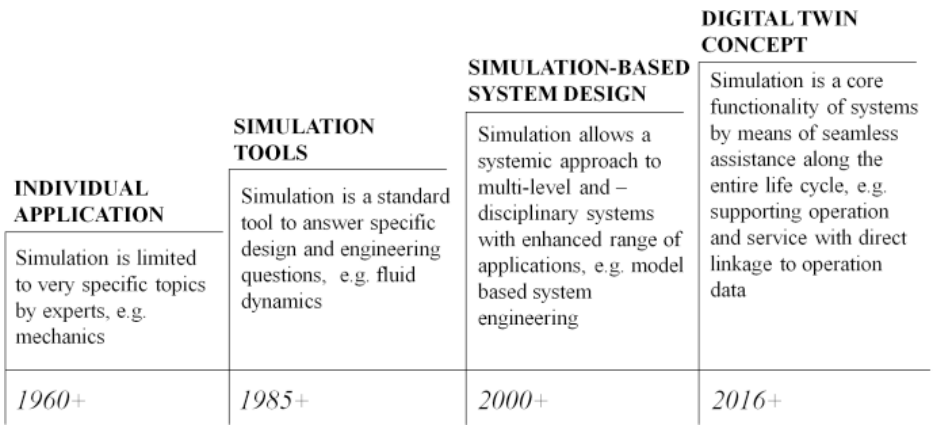


Figure 5: Digital simulation concept evolution

Every prospective of physical element, starting from a single machine to the whole production system, will be accompanied by virtual representations, which allow monitoring and predicting its performance in the physical world. The digital avatar will show at any time its current condition and information created by the real system along its own value chain and this information will dynamically grow during its lifetime, made possible by the increasing digitalization enabled by IoT and Big Data technologies in every stage of manufacturing. With the industry 4.0 initiatives, a real-time synchronization of the sensed data coming from the field to feed the DT should be made possible, which makes the simulation technology undergo a substantial change, the evolution of the concept is repesentes in Figure 5.

For most of the authors, DT is a model that represents the system on which different types of simulations can be based Meanwhile, in Robotics field, simulations are mainly performed for the Virtual Commissioning to optimize the control algorithms of robots during development phase, while in manufacturing, the main objective of simulations is to represent the complex behaviour of the system, also considering the possible consequences of external factors, human interactions and design constraints.

2.3.5. Are there any synonyms of "Digital Twin"?

Several synonyms have been identified in the scientific literatures: Digital Thread, Digital Counterpart, Virtual Twin, Product Avatar (PA), and Virtual Factory (VF). The term "Digital Thread" is usually addressed together with "Digital Twin", since the focus is not only a single phase such as engineering phase in general , but its whole product lifecycle from design, operation, usage to the final dismissal, this thread is connected. All these terms represent a virtual version of the physical one even if it is called in a different way. In this paragraph, the similarities and differences between these

keywords are specified. In addition, a few of them have been chosen to conduct the review together with “Digital Twin” and the results will be compared.

In terms of similarities, they are all representations of the physical product or system that include design specifications and engineering models describing its geometry, materials, components and behaviour, which enable us to better understand and design the real-world entity aiming to improve efficiency and reduce costs [29].

In terms of differences, DT is specific as it includes the as-built and operational data unique to the specific physical asset that it represents, it enables a real-time synchronization with the real system through the whole product lifecycle and it is under the industry 4.0 context. While the other terms do not necessarily have the same seamless real-time characteristic. PA describes a distributed and decentralized approach to share and manage the relevant, item-level information throughout a product‘s lifecycle (Beginning of Life (BOL) - Middle of life (MOL)- End of life (EOL)) among different shareholders [30]. By considering data from different life phase altogether will improve future product generations. It is very similar to DT, but it refers to only product level and it is mostly addressed in product lifecycle management (PLM) studies. Digital Thread is a continuous, seamless strand of data that connects each stage of the product life cycle from design, to build, to in field usage. It provides a formal framework for data with the ability to store, access, integrate, transform, and analyse from disparate systems throughout the product lifecycle into actionable information and drives efficient supply chain communications. DT is a “living” model with the purpose to monitor and optimize the physical performance. The Digital Thread enables the DT to fulfill this purpose by providing the kinds of data that the twin needs to perform its analyses. It coexists with the DT. Virtual Factory instead refers to an integrated model that involves a variety of software, tools, and methodologies in order to solve any real-time problem of manufacturing systems. It is more like a virtual commissioning tool and it performs virtual simulation exercises that helps in replicating the real-life scenario and helps in designing and implementation in a efficient and effective way. It does not depend on the pilot plants or production runs and does not need to connect with the real physical system. Everything is done on software.

Three similar terms have been chosen to be searched following the same search steps as the ones performed for the “Digital Twin” in the same three databases: Digital Counterpart, Product Avatar and Digital Thread. Since

these keywords might be more familiar in the research world and have been brought up earlier, the searching span has been put from year 2005.

2.3.6. What are the key enabling technologies for the "Digital Twin"?

As previously mentioned, this review focuses on the manufacturing domain, from product to production system level. All papers not related to this area were eliminated from the review. The key enabling technologies were extracted and summarized from the review papers, as shown in Figure 6.

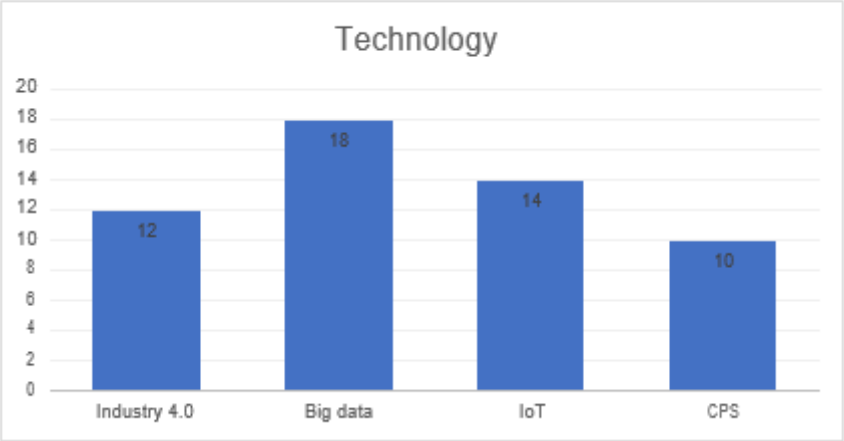


Figure 6: Driving technologies of the DT

As illustrated in the Figure 6, among these articles twelve of them are under context "Industry 4.0". The "Industry 4.0" paradigm was proposed for the first time at the Hannover Fair in 2011 [31], it stands for the fourth industrial revolution. It is known that after the 3rd industrial revolution, digitalization and computerization have been achieved and widespread. On the basis of an advanced digitalization within factories, the combination of internet technologies and future-oriented technologies in the field of "smart" objects (machines and products) seems to result in a new fundamental paradigm shift in industrial production. "Industry 4.0" represents the current trend of automation and data exchange.

"Industry 4.0" describes an exceptional technology-push paradigm in industrial practice, and it is mainly driven by Cyber Physical System, Internet of Things and Big Data. These technologies are integrated with the simulation model along the lifecycle to achieve increased competitiveness (energy and resource efficiency, shorter time to market, enhanced flexibility). With the use of these technologies, Industry 4.0 opens the way to real-time monitoring and synchronization of the real world activities to the virtual space thanks to the physical-virtual connection and the networking of CPS elements.

Cyber Physical Systems(CPS) are defined by the US National Science Foundation as "engineered systems that are built from, and depend upon, the seamless integration of computational algorithms and physical components. Advances in CPS will enable capability, adaptability, scalability, resiliency, safety, security, and usability that will far exceed the simple embedded systems". To put it in a simple way, CPS indicates a smart and self-aware machine where the physical and the digital level merge through a common communication layer. It has not only static information such as geometry, 3D model and bill of materials but also dynamic behaviour information which updates through the lifecycle.

The papers in the manufacturing field that have mentioned the use of the DT to simulate a CPS system or product are [32, 23, 21, 22]. Nowadays, CPS is still in the initial stage of development, it is essential to clearly define the structure and methodology of CPS as guidelines for its implementation in industry. In general, a CPS consists of two main fundamental functionalities: data acquisition and data management.

In 2015, a CPS 5C (Connection-Conversion-Cyber-Cognition-Configuration) level architecture was proposed in [33], to reach the goal of resilient, intelligent, and self-adaptable machines:

- *Connection*: data gathering from sensors and production control systems;
- *Conversion*: translation of relevant data acquired into consistent information;
- *Cyber*: modelling of physical elements;
- *Cognition*: simulation, interaction with humans and diagnostic;
- *Configuration*: active intervention in the real environment.

Internet of Things (IoT), also known as Internet of Objects, is an "emerging global Internet-based information architecture facilitating the exchange of goods and services, with the function to overcome the gap between objects in the physical world and their representation in information systems". In the CPS manufacturing context, all CPS units are interconnected and integration of sensors and actuators is more and more common, which leads to a highly distributed network of devices communicating with human beings as well as other devices. RFID (Radio-Frequency Identification) is one of the dispensable technologies for IoT, in which objects are tagged for their identification. Besides, sensors or other technologies that control remotely across existing network infrastructure can realize IoT. Digital exchange of data and information is made possible even without human intervention and it is not limited geographically. Thus, enormous amounts of data are

collected and gathered through sensors spread all over the factory, and need to be elaborated and transformed into actionable information through proper means. In the performed analysis, 14 papers talked about IoT in relation to DT. Canedo Strongly emphasized the importance of IoT and of DT as a new mechanism to manage IoT devices and IoT systems-of-systems throughout their lifecycle.

Big data is a loosely defined term to describe "data sets so large and complex that they become awkward to work with using standard statistical software". Eighteen papers among the analysed ones have mentioned in a implicit and explicit manner this technology. It is characterized with 3Vs: volume, velocity, variety. IoT makes the world more connected and networked, leading to a further large scale of data sets. In this information era, the way how useful information can be extracted and used for decision making is extremely crucial. As stated before, CPS utilize a huge amount of sensors and physical systems to collect sensory information from the real world and send them to the DT computation modules through communication technologies (e.g. wireless) and DT computation modules process these data and feedback to the physical systems about the findings, sometimes even send control commands to make necessary changes in the physical world or reconfigure system parameters if required. Big data and the IoT usually work in conjunction, data extracted from IoT device provides a mapping of device interconnectivity. Such mappings have been used widely by the media industry, companies and governments. But, it is also increasingly adopted in manufacturing contexts to gain better performance. Even though the Big Data topic is not strongly recognized by the authors as a key aspect in the DT modelling, as sensors are being progressively used in production system, it is impossible not to deal with a huge amount of data.

All the facilities in the physical world are interconnected through IoT, smart components with sensors gather real time data to enable seamless integration with the physical asset (Figure 7). The sensed data are send to the cyber twins of the physical assets, these cyber twins are also networked and interconnected such Big Data analytics are embedded in the cyper level and will process and transform the raw operation data into actionable insights then feed back to the physical machine through actuator or other control devices. All these integration of technologies are under the Industry 4.0 initiative. This is the journey of interactivity between the physical and digital worlds. Such a journey underscores the profound potential of the DT: thousands of sensors taking continuous, nontrivial measurements that are streamed to a digital platform, which, in turn, performs near-real-time analysis to optimize a business process in a transparent manner.

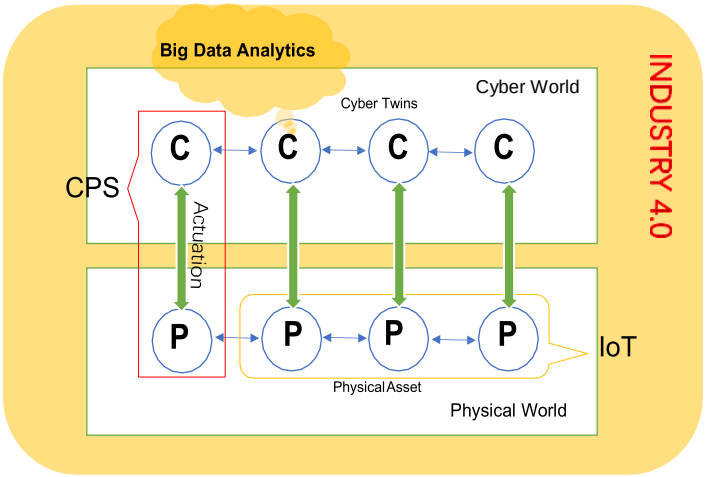


Figure 7: Integration of enabling technologies

2.3.7. Benefits of the Digital Twin

Thanks to Industry 4.0, IoT and Big Data, CPS are gradually replacing the traditional machines, DT is now becoming a reality. It will be definitely seen as the game changer in the future manufacturing. The information provided to these DTs by the physical product is providing manufacturers with more insight into their products and making it possible for them to predict more than they ever could before. Several benefits can be concluded in this literature review.

The main benefit of the DT is to provide an integrated outlook of any project, to any shareholders, at any point of the product lifecycle. Moreover,

1. As mentioned before, the concept of DT is not limited to a single product, but can be extended to complex system level.
2. The DT has the potential to radically change the design, manufacturing, sales, and maintenance of complex products in multiple industries. It makes the manufacturing system become more autonomous and self-aware and deliver the best business outcome in terms of time and cost [23].
3. With real-time mirroring, DT is not limited in the design engineering phase, but also play an important role along the whole lifecycle of a product or process. Information created in each stage of the product lifecycle is seamlessly made available thanks to DT, which makes it an important tool in PLM [23].
4. DT enables the interoperability of different systems and applications from different disciplines, engineers can simulate the behaviour of complex systems in order to predict and prevent mechanical breakdowns [26].

2.4. Industry 4.0 Lab at PoliMI

2.4.1. Lab capabilities

The developed DT simulation model was built to be validated at the Industry 4.0 Lab, as it was a laboratory with all the necessary features that were needed to test the developed model (i.e. connection with the field, detailed energy sensors, machine states reporting. . .). In this section the Industry 4.0 Lab ((I4.0 Lab) will be presented (Figure 8).

Industry 4.0 Lab is promoted by the Manufacturing Group of the Department of Economics, Management and Industrial Engineering (Dipartimento di Ingegneria Gestionale) of Politecnico di Milano and developed by FESTO1 and SIEMENS2. It aims to create a physical entity where the research activity in the innovative manufacturing management and planning approaches can be carried out in conjunction with a practical implementation in a "real-like" environment.

For the exploitation perspective, it is expected that I4.0 Lab could address 3 main purposes:

- Education and Training
- Communication and consulting
- Research Projects and related Activities

The manufacturing ecosystem includes 3 major components to consider: the human factor, the product and the process. These three factors need to be considered in a holistic way, in order to adequately build engineering and management competences. Industry 4.0 Lab has been built to address these three aspects in a flexible environment, to academically demonstrate the "Industry 4.0" vision. The covered technologies in this lab includes:

- CPS (Cyber-pysical system): Permits the intelligent networking of people, products, and production resources.
- RFID (Radio Frecuency Identification): Uses radio transmission to write or read data from labels without contact.
- NFC (Near Field Communication): Enables communication between two elements located close to each other.
- Cloud Technology: Stores sensors or data in cloud, and analyses Big –Data

To this end, an automated assembly line with a proper tracking and tracing solution, integrated with a storage area has been installed in the lab. This production system has been designed for an assembly of a Mobile phone. It consists of seven workstations, six CP Labs and one CP factory (CP Labs are: 1, 2, 3, 5, 6, 7; the CP Factory is: 4 (Figure 8)), CP stands for cyber

physical, they form the research platform for Industry 4.0. The difference between CP lab and CP factory is that CP lab has only one application module, and the size of the module is relatively limited while CP factory can contain multiple modules. The basic elements of these CPS are: Band, PLC, Application module and HMI (Human Machine Interface). CP-Bridge is used to combine CP-Factory and CP-Lab. No inter-operational buffer is presented and motorized conveyors serve as the handling system.

Two computers are equipped in the lab, one contains the Manufacturing Execution System (MES) for the generation and control of production orders and the other one has applications such as simulators and energy consumption monitoring software. Both computers use Windows Access as database, it is possible to configure the database to store the sensor values. All the facilities in the lab are connected to a network and each is assigned with an IP address.

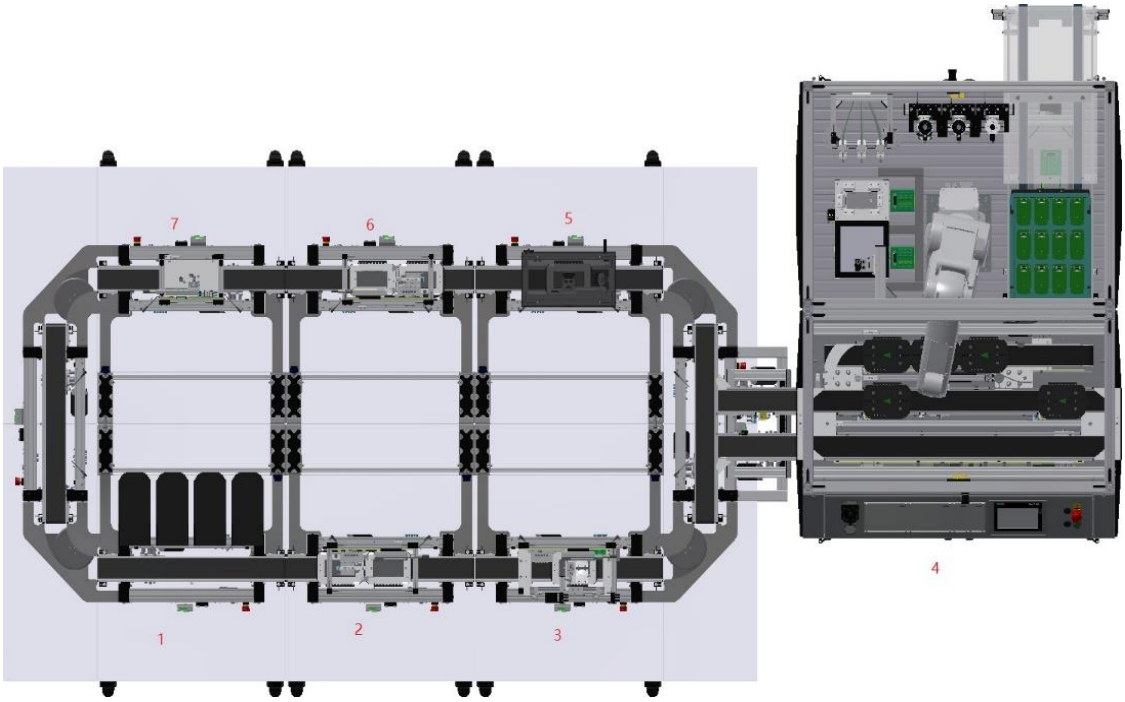


Figure 8: Industry 4.0 Lab Layout

1-Manual Module: Finished product is available to be unloaded from line, or an empty pallet is uploaded.

2-Base Magazine Module: A base is placed on an empty pallet.

3-Drilling Module: Drilling is executed

4-Robot Cell: It's the only CP-Factory, a PCB (Printed Circuit Board) is placed in the base and fuses are assembled on the PCB.

5-Visual Inspection Module: Visual inspection is carried out to check the

presence of proper fuse number.

6-Cover Magazine Module: A cover is placed on top.

7-Press Module: Cover is pressed onto the Base to close it.

2.4.2. The Product

As mentioned, this production system has been designed for an assembly of a Mobile pone. In this system, carrier and pallet are used as the support of the part. There are two types of sensors embedded in the carrier: screw sensor and RFID. These sensors communicate with the MES in order to proceed with the production. The system is completely controlled by the MES except the manual station where it requires one operator to unload the finish part. When a new production order is launched, it executes automatically following the steps to finish the assembly: 2 – 3 – 4 – 5 – 6 – 7 – 1(Figure 8). The product contains four elements, one front cover, one PCB, fuses and one back cover, the components are respectively labelled as 1, 2, 3, 4 (Figure10). This is the basic product that can be produced by the system (Figure9). Since this is a flexible and smart production system, the user can configure whatever desired product based on the available elements in the MES.

Figure 9: Final Product

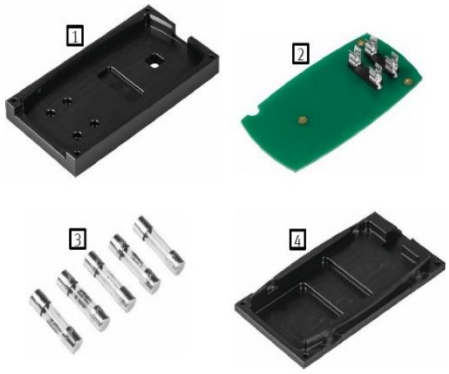


Figure 10: Product Elements

2.4.3. OPC Unified Architecture

Industry 4.0 Lab adopts OPC Unified Architecture (OPC UA), a Machine to Machine (M2M) communication protocol for industrial automation developed by the OPC Foundation, as its standard communication protocol. It plays an important role in the Industry 4.0 initiative as it helps getting one step closer to establishing a robust model based communication protocol between machines. It is also the key element to realize the IoT. The communication protocol complies with the IEC 62541 standard. Distinguishing characteristics are: Focus on communicating with industrial equipment and systems for data collection and control.

Using a common communication channel to access different powerful software tools will further extend the progress of establishing a unified architecture where different fields can be connected. This protocol focuses on communicating with industrial equipment and systems for data collection and control. Moreover, having widely used numerical computing environments and engineering tools like MATLAB and Simulink connected to the communication layer using client/server communications infrastructure, will expand the possibilities of merging different applications in Industry 4.0.

There are several open source implementations of OPC UA in different languages and with different licenses, such as open62541 based on C language. OpenOpcUa based on C++. These implementations support both server/client creation. While there are implementations that only support a client instance, like C# based opc-ua-client, the one used in Matlab and C++/Python based uaf. OPC UA supports two different protocols: The binary protocol is opc.tcp://Server and http://Server is for Web Service.

A meta model is defined to provide the infrastructure of OPC UA. This meta model presents nodes and references as its fundamental elements. A node class comprises other nodes. Depending on the node class, different attributes are defined in which some of them are mandatory and some are optional. References realize the relationships between the different nodes. Each OPC UA server has an address space where all the nodes are created. Each node has different attributes and each attribute has a value entry. Furthermore, an attribute could be a structure that has more attributes linked to it. OPC UA supports 26 built in data types. On top of that, what characterizes OPC UA is that it has its own defined security mechanism, thus an extra encryption layer implementation is not needed.

CHAPTER 3. TECHNIQUES, TOOLS AND METHODS BACKGROUND

3.1. IDEF() For representation for flow charts and process

In this thesis it has been decided to exploit a particular tool to represent the flow charts and the complex process of the thesis: IDEF. IDEF acronym stands for Integrated DEFintition: it is a method designed to model the decisions, actions, and activities of an organization or system. This methodology's peculiarity is that the process flows are representable in a multi-echelon structure. Starting from a broad process vision, according to the flow chart detail degree, this method allows to conduct the analysis deeply through the steps that compose the process. The two primary modeling components are: functions (represented on a diagram by boxes), and data and objects that interrelate those functions (represented by arrows). Starting from the very first level of aggregation, in which only one box is presented (IDEFØ), it is possible to appreciate its composition: the boxes that compose IDEFØ (Idef 1 level). In the detail is even possible to go more in the deep with the analysis, exploding the Idef1 level, comprehending its composition through the lower level Idef2, and so on according to the process detail.

Therefore, IDEF methodology exposes the process levels and compositions but also the interdependence between the sub-processes. Besides, one of the most important contribution of this method exploitation is the opportunity of both realizing the structure of a process and observing the Input, Output, Resources and Constrains involved in a particular process. The "box and arrow" graphics of an IDEFØ diagram show the function as a box and the interfaces to or from the function as arrows entering or leaving the box. The basic syntax for an IDEFØ model is shown in the Figure 11. IDEFØ is useful in establishing the scope of an analysis,

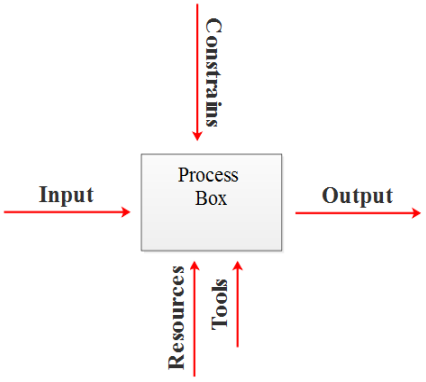


Figure 11: Idef0 representation conceptualization

especially for a functional analysis. As an analysis tool, IDEFØ assists the modeler in identifying what functions are performed, what is needed to perform those functions, what the current system does right, and what the current system does wrong. Thus, IDEFØ models are often created as one of the first tasks of a system development effort. Input: information that the process need in order to elaborate an output. It can be composed by whether information or materials that have been originated by a sub-process within the whole process or from an external source.

- Input: the information needed to deal with the elaboration process, available whether from the external environment or from the internal facilities or from both the sources (i.e. parts list or parts forecasting demand);
- constrains: as in every business field, there are features of the system that must be respected, whose limitations involve the exploitation of the optimal resource utilization method (as instance: engineering information and business restrictions);
- resources & tools: instruments through which the input can be modified and managed, according to the constraints procured from scarce resources, in order to obtain the output;
- output: final result, what the process reached after the input managing through the process constrains and the resources availability and administration.

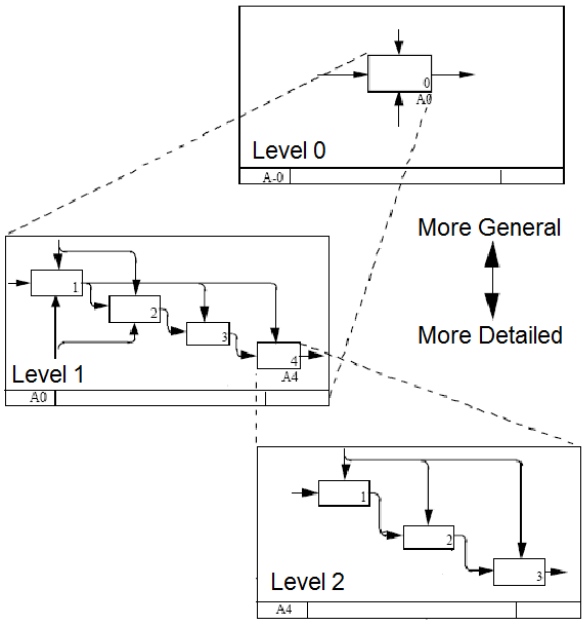


Figure 12: General Idef0 representation with number of level used in the framework.

In order to allow a clear and, as much as possible, easy interpretation of the connections that characterize the framework, a reading methodology is developed. How is showed in the figure below, each arrow is named accounting their box and

the side of the boxes from which they start and arrive. The boxes are identified by a number which characterized the position whether in the whole framework or in the exact process step. Besides, the boxes are also identified by a letter; the letter is useful in order to express the cause-effect connection through the boxes (Figure 12).

According to IDEFØ rules, the position at which the arrow attaches to a box conveys the specific role of the interface.

3.2. Matlab/Simulink OPC Toolbox for Data Acquisition

3.2.1. Communicating with PLCs.

Matlab/Simulink is one of the OPC UA client implementations using opc.ua.client. It is integrated in the OPC ToolboxTM. This ToolboxTM provides access to live and historical OPC data directly from MATLAB and Simulink. It allows reading, writing, and logging OPC data from devices, such as distributed control systems, supervisory control and data acquisition systems, and programmable logic controllers (PLC). OPC Toolbox lets the user work with data from live servers and data historians that conform to the OPC Unified Architecture (UA) standard. Since all the modules in the I4.0 Lab are equipped with PLC and these PLCs are activated as OPC UA servers (Figure 13). Thus, by using the toolbox, it is possible to communicate in real time with the facilities. In the OPC UA client/server connection scheme, multiple clients can be created at the same time to query the servers.

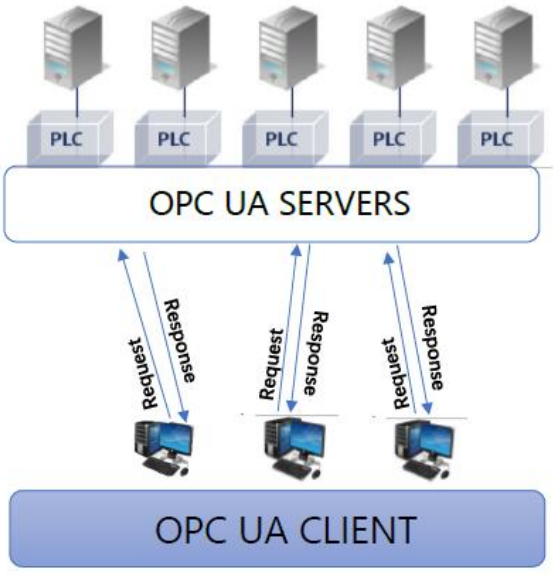


Figure 13: Connection by OPC UA between PLC and computer

In Matlab/Simulink, OPC UA client can be created to connect to OPC UA servers using the *opcua( )* function. After the client has been constructed, using the

connect() to connect the OPC UA server, it is based on the binary protocol thus the IP address of the server needs to be entered. From then on, it is possible to read whatever value available in the server using the function *readvalue( ),* the input argument of this function is the node of the desired data, an OPC UA Node variable describes the node on the server, and contains other subnodes in the 'Children' property. Nodes have a NodeType which can be 'Object' or 'Variable'. Object nodes have no value associated with them, and are used purely for organizing the namespace of the server. Variable nodes store current values, representing a sensor or actuator value associated with the server. But unfortunately, in Simulink workspace, the OPC toolbox library does not support the OPC UA server connection as it is a new protocol, a new way to implement the connection in Matlab need to be found. One way to do that is by using an classical Matlab Function block in simulink clarifying the OPC UA functions descried above as intrinsic functions, but the logic of this Matlab function turns out to be that the function will be called multiple times during the simulation and every time the connection process will be replicate, a certain delay would present or loss of data in a more accurate sense. In order to increase the accuracy of the data acquisition, a new way to connect only once the server and then looping just the function *readvalue( )* during the simulation run has to be found. In this way, a real time synchronization can be achieved. In this regard, another way is to implement real time data acquisition in Simulink with level 2 S function. This function permits the connection in setup function and saves the client nodes in memory, then the memory will be retrieved in the output function to get the value. During the simulation, only the output function is executed.

3.2.2. Matlab level 2 S-Function for Data Adquisition

Apart from the connection OPC UA servers by the program UAExpert, there is an important MATLAB Simulink tool that has a relevant role on the modelling: the Level 2 MATLAB S-Function. The Level-2 MATLAB S-function allows to use the MATLAB language to create custom blocks with multiple input and output ports and capable of handling any type of signal produced by a Simulink model, including matrix and frame signals of any data type. The Level-2 MATLAB S-function corresponds closely for creating C MEX S-functions. To avoid duplication, this section focuses on providing information that is specific to writing Level-2 MATLAB S-functions. A Level-2 MATLAB S-function is MATLAB function that defines the properties and behavior of an instance of a Level-2 MATLAB S-Function block that references the MATLAB function in a Simulink model. The MATLAB function itself comprises a set of callback methods that the Simulink engine invokes when updating or simulating the model. The callback

methods perform the actual work of initializing and computing the outputs of the block defined by the S-function.

To facilitate these tasks, the engine passes a run-time object to the callback methods as an argument. The run-time object effectively serves as a MATLAB proxy for the S-Function block, allowing the callback methods to set and access the block properties during simulation or model updating.

The Level-2 MATLAB S-function defines the signatures and general purposes of the callback methods that constitute a Level-2 MATLAB S-function. The S-function itself provides the implementations of these callback methods. The implementations in turn determine the block attributes (e.g., ports, parameters, and states) and behavior (e.g., the block outputs as a function of time and the block inputs, states, and parameters). By creating an S-function with an appropriate set of callback methods, it can be defined a block type that meets the specific requirements of the application requested.

A Level-2 MATLAB S-function must include the following callback methods:

- A setup function to initialize the basic S-function characteristics
- An Outputs function to calculate the S-function outputs

The S-function can contain other methods, depending on the requirements of the block that the S-function defines. The methods defined by the Level-2 MATLAB S-function generally correspond to similarly named methods defined by the C MEX S-function.

3.3. Asset Health Index basis for Machine Condition Assessment

Asset Health Indicators or Indices (AHI) are widely used in supporting maintenance and replacement strategies based on asset condition and performance. This indicator is normally a composite metric intended to provide information of asset capability over time. The Asset Health Index indicates the health of an asset and is an aggregation of the measurement results of the condition criteria of an asset and a weighting factor for each of those measurements. . For a successful AHI we need to link the available raw data– whether condition monitoring or asset history or maintenance and operational data - through to likely failure modes, or issues which will affect asset performance [34].

When appropriately developed, health indices provide an accurate indication of the probability of asset failures and associated risks. Having established the asset health index under current conditions, health index values in future can be predicted by taking into account the impact of environmental and operating conditions along with the preventative maintenance practices.

Health indexing quantifies equipment condition based on numerous condition criteria that are related to the long-term degradation factors that cumulatively lead to an asset' end-of-life. Health indexing results differ from maintenance testing, which emphasizes finding defects and deficiencies that need correction or remediation to keep the asset operating during some time period [35].

3.3.1. Background and basic definitions.

The method to be used in this work takes into account several features of well known assets health Indexing models with the intention to reach a practical procedure that can be applicable to any complex asset, regardless its technology, industrial sector, or location where it is used.

This procedure takes as a basic reference for its elaboration the model presented within a so-called framework of common reference presenting the principles and calculation methodology adopted by all British power network operators for the regulatory assessment, prediction and report of the risk of assets. This framework main figures considered in this work are:

- *The asset health index* (AHI) is considered as a dimensionless number between 0.5 (which corresponds to its status or condition as new equipment) and the value of 10 (corresponding to the condition of the equipment at the end of its useful life). The behavior pattern of the AHI, is supposed to be exponential along the age of the asset. The following figure shows the different 5 sections into which the health index of the asset is divided (Figure 14).

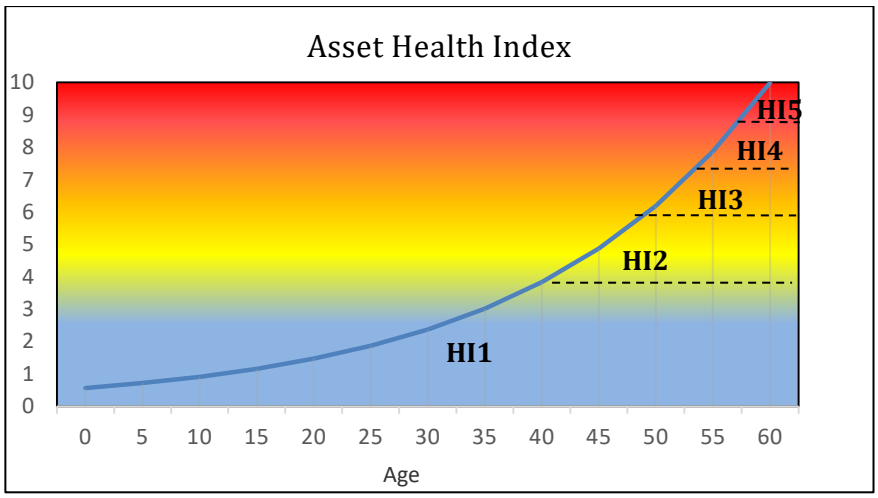


Figure 14: AHI Intervals.

The HI1 range comprised 0.5≤AHI≤4 values; for which the behavior of the equipment is assumed to resemble as new equipment. The HI2 range

considers AHI values within the interval 4<AHI≤6 and corresponds to the period of time when the first signs of deterioration begin to appear in the equipment. In this range, the value corresponding to AHI=5.5, is assumed to be the health index value equivalent to the normal life expected for the equipment category. From this point, three intervals are considered in the methodology: HI3, HI4 and HI5 as the AHI exceeds the values of 6, 7 and 8 respectively. The methodology assumes that exceeded the value of AHI=8, the equipment is at the end of its useful life.

- *The asset location factor.* This is a factor is considered to be inherent to the functional location of the equipment, in the facility and the geographical area where it is located. Exposure to environmental agents, whether or not they are protected from the outside, distance from the coast or working at a certain altitude will have a different effect on the health of the equipment. In this work the asset location factor does not have special impact on health.
- *The asset load factor.* The load factor of each equipment is defined by the relationship between the real load of the equipment at its expected operating point or schedule, and the maximum admissible load that the equipment could support. The load factor is therefore a consequence of the operating conditions for which the installation has been designed.
- *Health and reliability modifiers.* Health modifiers are parameters that contain additional information about current health of the asset. The health modifiers are related to current load, condition of the asset (result of inspections and checks performed on it), and operation (result of captures made of operational variables of the asset, for example those existing in the plant information systems). Reliability modifiers apply to assets for which reliability can significantly differ within the same category. In our model, these modifiers will have no relevant impact at this stage of the modeling process. The different reliability modifiers can be associated with:
    - The manufacturer brand and the type or model of asset.
    - The construction and integrity of the asset, the material, etc.
    - Surface applied treatments.
    - Number of overhauls and large maintenance performed on the equipment.
    - Hours of inactivity of the equipment.
    - The use of original spare parts
    - Etc.

3.3.2. Data requirements.

Data requirements for the procedure to be implemented will be as follows:

- *The identification of the asset*. The category of the asset, the current age, the expected lifetime, name of the manufacturer, the model and functional location in the company.
- *Operation & Maintenance data.* Recorded during a certain period of time, added conveniently to know the time in which the equipment has been subjected to stress, number of starts and stops, energy consumption, number of overhauls, etc.
- *The condition of the equipment*. The results of the analyzes performed on the equipment in situ, results of readings of physical variables such as temperature and vibrations, results of visual inspections, insulation, thermography, etc.

3.3.3. Procedure.

The methodology in this paper is based on a procedure consisting in 5 consecutive steps. Starting from an estimated normal life associated with a category of equipment, the current health index of an asset is reached. This procedure is presented in Figure 15.

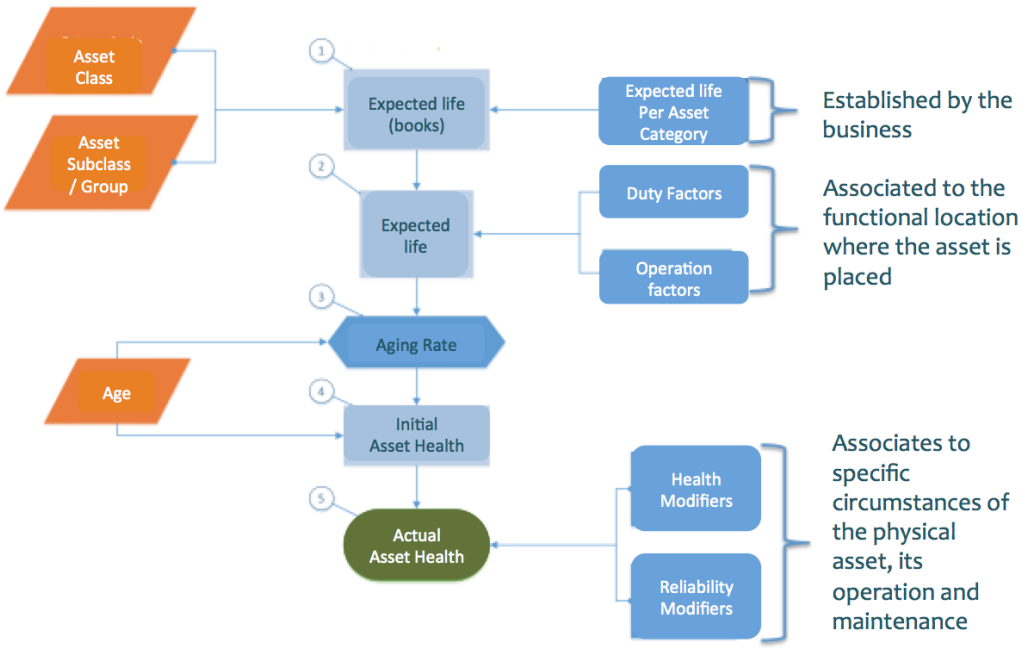


Figure 15: Procedure to calculate the AHI

Step 1. Asset selection and category definition. Capture of functional location data, physical asset data and obtaining the estimated normal life of the asset.

In this first step, the identification of the asset and all the information regarding its functional location is addressed. Regarding the definition of the category of

asset, this procedure should make clear the category/subcategory of the assets involved. For instance: Drilling machine/electrical.

Once the assets conveniently identified and classified, the following is to obtain all data required to characterize the functional location where the equipment is located. Regarding the physical asset data, equipment information like manufacturer, model and technical design specifications will be required.

The estimated normal life of the asset is a data that is generally provided taking into account the experience accumulated so far and the information provided by the different manufacturers. For many equipment, the concept of estimated normal life is replaced by a succession of time periods that run until the overhaul of the equipment (this happens for instance in equipment like turbochargers) or until the realization of the so-called major maintenance operations (like in big pumps and compressors). After the realization of each overhaul and major maintenance, the equipment is considered many times as new, in practice this means placing the value of the health index back to 0.5.

The value of the estimated normal life is used as a starting point for the realization of all the calculations that will be seen below. Keep in mind that its value is and approximated value and only depends on the asset category. As we will see below, it will be modified by asset functional location and loading.

Step 2. Evaluation of the impact of location and load factors by type of asset, technical location and estimated life expectancy

Once all the information in the previous point has been compiled, the location and loading factors are evaluated (unambiguously associated with the asset's technical location, as previously mentioned) and impact on health estimated (in this work in equivalent hours) the estimated life of the asset (or the time to the noverhaul) must be defined.

Step 3. Calculation of the aging rate

A fundamental hypothesis of the chosen methodology is that the aging of an asset has an exponential behavior with respect to its age. The aging rate is the parameter of the model that allows us to express mathematically this mode of behavior, and that incorporates the different phenomena that the asset can suffer throughout its useful life, such as corrosive phenomena, wear, oxidation of oils, breakage of insulation, etc. The aging rate (β) it is determined by the relationship between the natural logarithm of the quotient between the health corresponding to the new asset and the health the asset will have when reaching the estimated life, and the calculated estimated life (See Equation 5).

$$\beta = \frac{\ln \frac{HI\ new}{HI\ estimated\ life}}{\text{Estimated Life}} \quad (5)$$

Where:

B :	Aging rate.
HI_{new} =0.5	Health index corresponding to a new asset;
$HI_{estimated\ life}$ =5.5	Health index corresponding to an asset arriving to its estimated life;

Step 4. Obtaining the Health Index at the age t

The health index, as mentioned before is a dimensionless number between 0.5 and 10, with an exponential behavior with respect to the age of the asset (t), which is characterized by the aging rate of the equipment. For the calculation of the health index (HI) of an asset, Equation 5 is used, where t is the current age of the asset (in units of time) and the aging rate β is calculated in step Equation 6.

$$HI(t) = HI_{new}\ e^{\beta t} \quad (6)$$

Figure 16 represents the evolution of the initial health index of an asset with an estimated normal life of 50 months, with a factor of location and load inherent to a particular technical location.

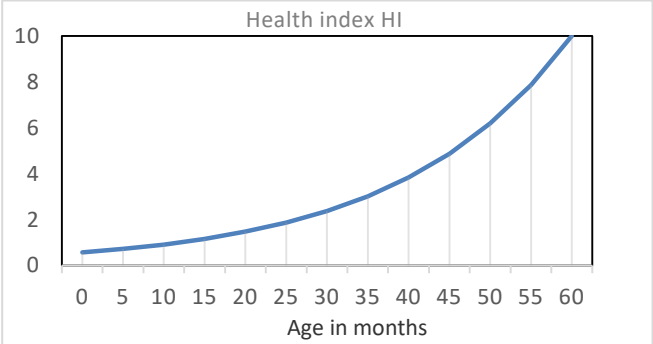


Figure 16: Graph of the health index of an asset (*HI*).

Step 5. Evaluation of the probability of failure

The methdology adopted assumes the probability of failure is a polinimial (normaly 3rd order) or exponential function of the asset´s health index. Once the HI increases over 4, the probability of failure will increase from normal failure rate to a maximum of 10 times this failure rate in case HI may reach 10.

3.4. Artificial Neural Networks (ANN) to Model Complex Behavior

In the development of the final step of the thesis it is used an artificial neural network, in order to have a deeper analysis of the fail in the machine based on the asset´s condition and health. An artificial neural network (ANN) is a computational

model based on the structure and functions of biological neural networks. It is like an artificial human nervous system for receiving, processing, and transmitting information in terms of Computer Science. Basically, there are 3 different layers in a neural network (Figure 17):

- Input Layer (All the inputs are fed in the model through this layer)
- Hidden Layers (There can be more than one hidden layers which are used for processing the inputs received from the input layers)
- Output Layer (The data after processing is available at the output layer)

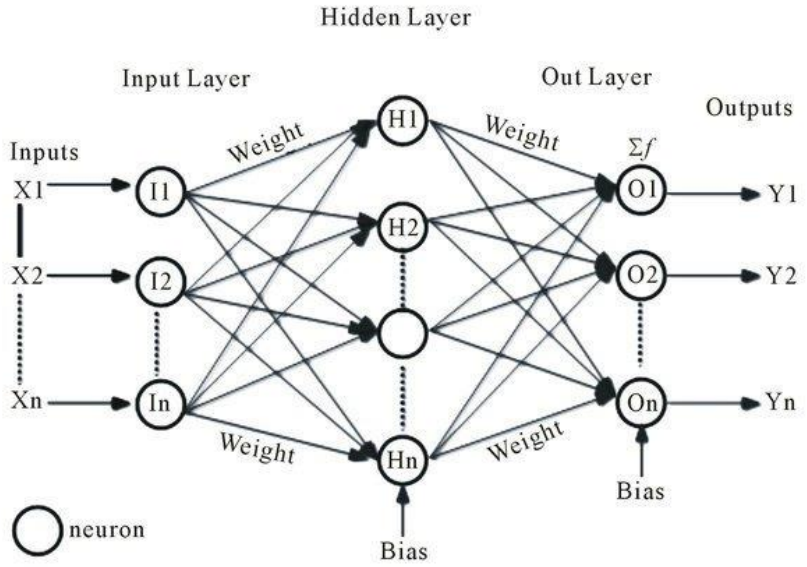


Figure 17: Different layers of a neural network

The Input layer communicates with the external environment that presents a pattern to the neural network. Its job is to deal with all the inputs only. This input gets transferred to the hidden layers which are explained below. The input layer should represent the condition for which we are training the neural network. Every input neuron should represent some independent variable that has an influence over the output of the neural network.

The hidden layer is the collection of neurons which has activation function applied on it and it is an intermediate layer found between the input layer and the output layer. Its job is to process the inputs obtained by its previous layer. So it is the layer which is responsible of extracting the required features from the input data. Research has been made in evaluating the number hidden layers (deep of the network) and the number of neurons in these layers. In case of problems dealing with complex decisions, up to 5 hidden layers, and up to 10 neurons are normally used based on the degree of complexity of the problem, or the degree of accuracy required. That certainly does not mean that increasing the number of layers or neurons will always give higher accuracy.

The output layer of the neural network collects, process and transmits the information in a way it has been designed. The pattern presented by the output layer can be directly traced back to the input layer. The number of neurons in output layer should be directly related to the type of work for which the neural network was designed, therefore, to determine the number of neurons in the output layer, first the intended use of the neural network must be considered.

The procedure used to carry out the learning process in a neural network is called the training algorithm. There are many different training algorithms, with different characteristics and performance. The learning problem in neural networks is formulated in terms of the minimization of a loss function, f. This function is in general, composed of an error and a regularization terms. The error term evaluates how a neural network fits the data set. On the other hand, the regularization term is used to prevent overfitting, by controlling the effective complexity of the neural network.

The loss function depends on the adaptative parameters (biases and synaptic weights) in the neural network. We can conveniently group them together into a single n-dimensional weight vector w.

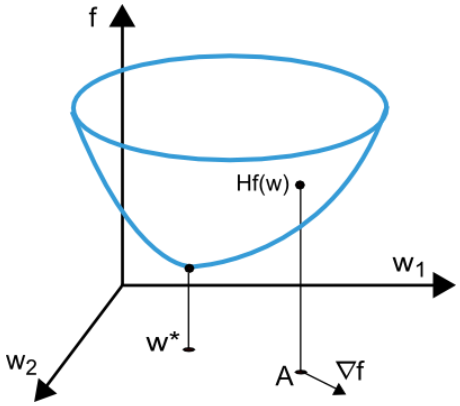


Figure 18: The representation of the loss function f(w) and the weights of the network

As we can see in Figure 18, the point w* is minima of the loss function. At any point A, we can calculate the first and second derivatives of the loss function. The first derivatives are grouped in the gradient vector, whose elements can be written as

$$\nabla_i f(w) = \frac{df}{dw_i}, \quad with \ i=1 \dots n$$

The problem of minimizing continuous and differentiable functions of many variables has been widely studied. Many of the conventional approaches to this problem are directly applicable to that of training neural networks.

The learning problem for neural networks is formulated as searching of a parameter vector w* at which the loss function f takes a minimum value. The necessary condition states that if the neural network is at a minimum of the loss function, then the gradient is the zero vector. The loss function is, in general, a non-linear function of the parameters. As a consequence, it is not possible to find closed training algorithms for the minima. Instead, we consider a search through the parameter space consisting of a succession of steps. At each step, the loss will decrease by adjusting the neural network parameters.

In this way, to train a neural network we start with some parameter vector (often chosen at random). Then, we generate a sequence of parameters, so that the loss function is reduced at each iteration of the algorithm. The change of loss between two steps is called the loss decrement. The training algorithm stops when a specified condition, or stopping criterion, is satisfied.

The most popular training algorithms for neural networks are: Gradient descent, Newton´s method, Conjugate gradient, Quasi newton and Levenberg Marquardt .

Gradient descent, also known as steepest descent, is the simplest training algorithm. It requires information from the gradient vector, and hence it is a first order method. This method will be used in this chapter to follow what is named a backpropagation process to train the network, optimizing the weights so that the ANN can learn how to correctly map or link arbitrary inputs to outputs [36][37][38].

3.4.1 The forwards pass

The ANN makes a prediction given certain weights and biases and certain inputs. Once these values are known, the total net input to each hidden layer neuron can be figured out, and then, using an activation function, the output of each neuron is computed. The process is repeated with the output layer neurons (Figure 19).

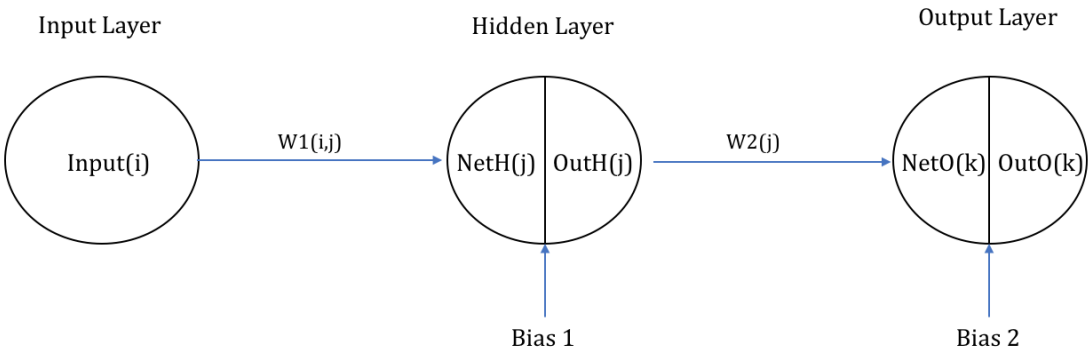


Figure 19: Simplified selected representation of variables in the neural network

For the purpose of this work, we will consider *j=1…m* number of neurons in only one single hidden layer, and *i=1…n* number of inputs.

Total net input for a hidden layer is defined as follows:

$$NetH(j)_t = \sum_{i=1}^{i=n} Input(i)_t \cdot w1(i,j)_t + Bias1 \quad (1)$$

The output of neuron *j* in the hidden layer will be:

$$OutH(j)_t = \frac{1}{1+e^{-NetH(j)_t}} \quad (2)$$

The function in Equation 2 is known as the activation function of the network, and the a bias value (*Bias1*) in Equation 1, allows us to shift the activation function to the left or right, which may be critical later for successful learning.

Once all hidden layer inputs are calculated the same process is repeated for the output layer neuron (assumed there is only one output in this example).

$$NetO_t = \sum_{j=1}^{j=m} OutH(j)_t \cdot w2(j)_t + Bias2 \quad (3)$$

Notice that, in this process, we use the outputs of the hidden layer neuros as inputs of the output layer neuron.

$$OutO_t = \frac{1}{1+e^{-NetO_t}} \quad (4)$$

3.4.2 Calculating the Total Error

The error of an output neuron, loss function to which we referred above, can be calculated using the square error function as follows:

$$Error_t = \sum_{k=1}^{k=K} \frac{1}{2} \cdot (Target(k)_t - OutO(k)_t)^2 \quad (5)$$

In this case with k=1, the equation can be presented as follows:

$$Error_t = \frac{1}{2} \cdot (Target_t - OutO_t)^2 \quad (6)$$

Where Target is the ideal output given the current inputs, and the ½ is included so that the exponent is cancelled when the equation is, later on, differentiated.

3.4.3 The Backwards Pass

The back propagation process goal is to update each of the weights in the ANN so that they cause the actual output to be closer to the target ideal output, thereby minimizing the error function for the output neuron.

The Output Layer: the back propagation process requires to know how much a change in w2(j) will affect the total error, for that purpose we can estimate the partial derivative

$$\frac{dE_t}{dw2(j)_t} \tag{7}$$

Once this is estimated the process will calculate the new weights as follows:

$$w2(j)_t = w2(j)_{t-1} - \eta \cdot \frac{dE_t}{dw2(j)_t} \tag{8}$$

Where η is considered learning rate. This parameter scales the magnitude of our weight updates in order to minimize the network's error function. If the learning rate is set too low, training will progress very slowly as you are making very tiny updates to the weights in the network. However, if the learning rate is set too high, it can cause undesirable divergent behavior in the loss function. We will see later in this Chapter how we can optimize values for learning rate and biases on a comprehensive manner.

Let's now obtain the previously referred partial derivative, in Equation 7, by applying the chain rule:

$$\frac{dE_t}{dw2(j)_t} = \frac{dE_t}{dOutO_t} \cdot \frac{dOutO_t}{dNetO_t} \cdot \frac{dNetO_t}{dw2(j)_t} \tag{9}$$

Now each piece of this Equation 9 will be figured out:

$$\frac{dE_t}{dOutO_t} = 2 \cdot \frac{1}{2} \cdot (Target_t - OutO)^{2-1} \cdot (-1) = OutO_t - Target \tag{10}$$

Then, applying the corresponding derivatives, it can be demonstrated that:

$$\frac{dOutO_t}{dNetO_t} = OutO_t \cdot (1 - OutO_t) \tag{11}$$

Finally:

$$\frac{dNetO_t}{dw2(j)_t} = \frac{d(\sum_{j=1}^{j=m} OutH(j)_t \cdot w2(j)_t + bias2)}{dw2(j)_t} = OutH(j)_t \tag{12}$$

Therefore, it can be notice that Equation 9, can be rewritten as follows:

$$\frac{dE_t}{dw2(j)_t} = (OutO_t - Target_t) \cdot OutO_t \cdot (1 - OutO_t) \cdot OutH(j)_t = \delta o_t \cdot OutH(j)_t \tag{13}$$

This in known as the Delta Rule, with

$$\delta o_t = (OutO_t - Target_t) \cdot OutO_t \cdot (1 - OutO_t) \tag{14}$$

Thus, Equation 8 can be expressed as

$$w2(j)_t = w2(j)_{t-1} - \eta \cdot \delta o_t \cdot OutH(j)_t \tag{15}$$

By doing so the actual updates in the neural network can be performed, after the new weights leading into the hidden layers neurons are found (i.e. the original weights, not the updated weights, will be used when continuing with the back propagation algorithm as described below.

The Hidden Layer: the backwards pass must continue by updating the values for $w1(i,j)_t$. In order to do so, the method requires to know how much a change in *w1(I,j)* will affect the total error, for that purpose we can estimate the partial derivative in Equation 16.

$$\frac{dE_t}{dw1(i,j)_t} \tag{16}$$

Once this is estimated the process will calculate the new weights as follows:

$$w1(i,j)_t = w1(i,j)_{t-1} - \eta \cdot \frac{dE_t}{dw1(i,j)_t} \tag{17}$$

By applying again the chain rule

$$\frac{dE_t}{dw1(i,j)_t} = \frac{dE_t}{dOutH(j)_t} \cdot \frac{dOutH(j)_t}{dNetH(j)_t} \cdot \frac{dNetH(j)_t}{dw1(i,j)_t} \tag{18}$$

Again, each piece of this equation must be figured out.

$$\frac{dE_t}{dOutH(j)_t} = \frac{dE_t}{dNetO(j)_t} \cdot \frac{dNetO_t}{dOutH(j)_t} \tag{19}$$

With

$$\frac{dE_t}{dNetO_t} = \frac{dE_t}{dOutO_t} \cdot \frac{dOutO_t}{dNetO_t} = (OutO_t - Target_t) \cdot OutO_t \cdot (1 - OutO_t) \tag{20}$$

And with

$$\frac{dNetO_t}{dOutH(j)_t} = w2(j)_t \tag{21}$$

Therefore

$$\frac{dE_t}{dOutH(j)_t} = (OutO_t - Target_t) \cdot OutO_t \cdot (1 - OutO_t) \cdot w2(j)_t \tag{22}$$

If we continue calculating the second product of Equation (18)

$$\frac{dOutH(j)_t}{dNetH(j)_t} = OutH(j)_t \cdot (1 - OutH(j)_t) \tag{23}$$

If the third piece of Equation (18)

$$\frac{dNetH(j)_t}{dw1(i,j)}_t = Input(i)_t \quad (24)$$

Therefore Equation (18) can be written as follows

$$\frac{dE_t}{dw1(i,j)_t} = (OutO_t - Target_t) \cdot OutO_t \cdot (1 - OutO_t) \cdot w2(j)_t \cdot OutH(j)_t \cdot (1 - OutH(j)_t) \cdot$$
$$\cdot Input(i)_t = \delta o_t \cdot w2(j)_t \cdot OutH(j)_t \cdot (1 - OutH(j)_t) \cdot Input(i)_t \quad (25)$$

And then Equation (25) can be expressed, using the Delta Rule, as follows:

$$\frac{dE_t}{dw1(i,j)_t} = \delta H(j)_t \cdot Input(i)_t \quad (26)$$

With

$$\delta H(j)_t = \delta o_t \cdot w2(j)_t \cdot OutH(j)_t \cdot (1 - OutH(j)_t) \quad (27)$$

Finally we can update the weights as follows:

$$w1(i,j)_t = w1(i,j)_{t-1} - \eta \cdot \delta H(j)_t \cdot Input(i)_t \quad (28)$$

And by doing so we finish the backwards pass and the backpropagation algorithm.

In this work a continuous simulation model of ANN with backpropagation will be later presented

3.5. Introduction to Continuous Time Dynamic Simulation

Dynamic mathematical models used in computer simulation are typically represented with differential equations (the relationship involving the rates of change of continuously changing quantities modelled by functions) or difference equations (relating a term in a sequence to one or more of its predecessors in the sequence). There is a clear reason for this that is related to the nature of the system being modelled. Some industrial systems or processes, like many process plant processes, occur continuously in time. Others, such as certain manufacturing processes, occur more discretely in time. Even though data collected from continuous processes are by necessity taken at discrete time intervals, model predictions based on these data assume temporal continuity and are commonly written in the form of differential equations. By contrast, discrete-time processes are modelled using difference equations, equations that take into account the discontinuous nature of these processes.

Difference equations are used in systems where change occurs at discrete points in time. Difference equations suppose that future values of variables of a system are a function of the current and possibly past values. For instance, a first-order difference

equation, given below, supposes that the next period value is only a function of the current period value:

$$x_{t+1} = f(x_t)$$

where $f(x_t)$ may be either a linear or nonlinear function, and the starting value x_0 is needed for the equation to be solved. A general k-order difference equation takes the form

$$x_{t+k} = f(t, x_t, x_{t+1}, \dots, x_{t+k-1})$$

Obviously, for a k-order equation we need $k-1$ starting values – x_0, x_1, …, x_{k-1} – to determine x_k. Again, $f(t, x_t, x_{t+1}, \dots, x_{t+k-1})$ may be either a linear or nonlinear function.

3.5.1. Stock and flow diagram

Stock and flow diagrams (SFD) — or level and rate diagrams (LRD) — are ways of representing the structure of a system and the typology of the variables selected for its representation.

Stocks (or levels) are fundamental to generating behavior in a system; flows (or rates) cause stocks to change. Stock and flow diagrams contain specific symbols and components representing the structure of a system. Stocks are things that can accumulate — such as inventory — and are represented with boxes.

Flows represent rates of change and they are expressed by decision functions — such as reductions in inventory through sales — and they are represented or drawn as valves. These diagrams also contain “clouds” (which represent the boundaries of the problem or system in question), auxiliary variables, etc.

Our models are composed of interconnected networks of stocks and flows, including many information channels, which connect the levels to the decision functions. Stock and flow diagrams are the most common first step in writing the executable code of a simulation System Dynamics model because they help to define types of the variables that are important in causing behavior. Therefore we can say that stock and flow diagrams provide a bridge from conceptual modelling to assigning equations to the relationships between variables.

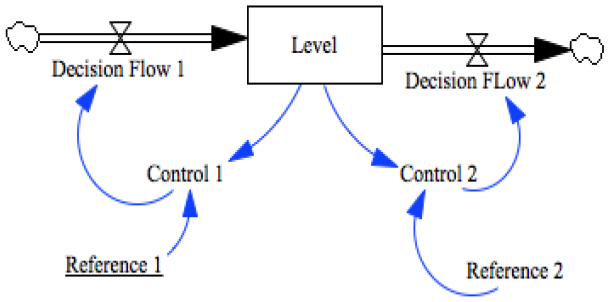


Figure 20: Sample Stock and Flow Diagram (SFD)

Figure 20 depicts a very simple structure of a reservoir or level, with an inflow and an outflow. To specify the dynamic behavior, a system of equations is defined. It consists of two types of equations, which correspond to levels and decision functions (rates). Equations control the changing interactions of a set of variables, as time advances. The continuous advance of time is broken into small intervals of equal length dt. For example the equations describing the state of the levels in Figure 4 is

$$Level(t) = Level(t - dt) + dt \cdot [Decision\ Flow\ 1(t) - Decision\ Flow\ 2(t)]$$

$$Level(t_o) = Level_o$$

Levels in Figure 20 at time t depend on its value at time t-dt and the value going in from decision function 1 minus the value going out to decision function 2. Notice that it is necessary to give the initial value of it to solve this equation.

There will be as many equations as variables. To determine the variables' behavior, the differential equations system is integrated. This can be easily done with Vensim.

CHAPTER 4. WORK PROCESS DESIGN USING IDEF()

4.1 The IDEFØ Diagram of the process

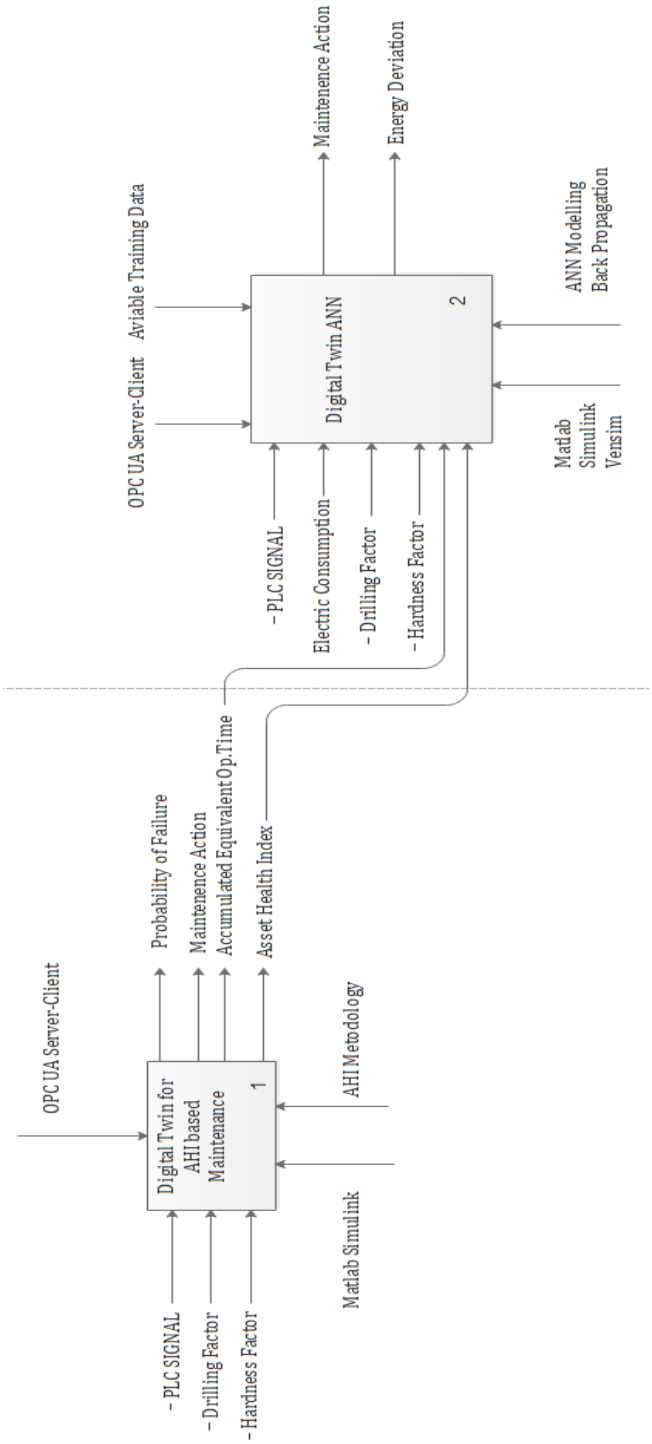


Figure 21: IDEFØ representation of the whole tesis work

In this chapter, once the state of the art is defined and a deep review of the DT concept is exposed, the central work of the thesis is developed. It is used IDEF()to represent the process that it is followed. The representation shows two levels of detail in order to present the work of the thesis as clear and estructurated as possible.

Firstly, the IDEFØ model (Figure 21) presents the cyber architecture part in two different dimensions. The first one represents the DT model creation and application to control and monitor the health and probability of failure of the asset. The box number 2, represents the further application and development of the ANN to have a powerful tool, which will have a wide range of purposes in Condition Based and Smart Maintenance. An intelligent maintenance system is a system that utilizes the collected data from the machinery in order to predict and prevent the potential failures in them. The goal of intelligent maintenance systems is to achieve and sustain near-zero breakdown. Such goal can be achieved by the transformation of raw data to valuable information regarding the current and future condition of the asset or process being monitored [39].

4.2 IDEF1 Diagram

The IDEF1 represents each of the two boxes in a deeper level of detail. In the box 1 (Figure 22), the Plant control outputs are the signals given by the PLC, which are the inputs for the monitoring at real time. Up to this point, the DT is the main tool to monitor the physical system state (Machine from the Industry 4.0 Lab), and make an historical time monitoring of the state.

The DT modelling has been developed in Matlab Simulink, with the SimEvents library and the Level-2 S- functions that will be explained below in the thesis. Thus, a simulation model focused on the machine degradation is developed. It is based on simulated degradation factors and the condition of the machine depending on the stress of the work in progress. Actually, the model will provide an “Asset Health Index” indicator, which allowes to monitor the machine condition and the probability of failure from the asset while working, as well as a preventive maintenance action when the probability of failure is bigger than the policy of failure defined.

To continue with the IDEF1 representation level, the second box of the IDEFØ diagram, which refers to the ANN is developed (Figure 23). It shows the three different phases that the process goes through to finally reach a fully trained and aviable ANN. The first step is the election the inputs and outputs that are going to form the ANN. Then it will be necessary to use the OPA UA to extract the value of the signals, and the Matlab Level-2 S-functions in order to convert the signals into training model variables. Once the data to feed the ANN is obtained, the second step

of the process begins. The ANN training is made using Vensim model, that allowes to implement Backpropagation algorithm. So after the training with the inputs and outputs carefully chosen, the parameters that conform the ANN get calibrated. When that happens the ANN is aviable to give an output at real time from the inputs that it gets by the PLC.

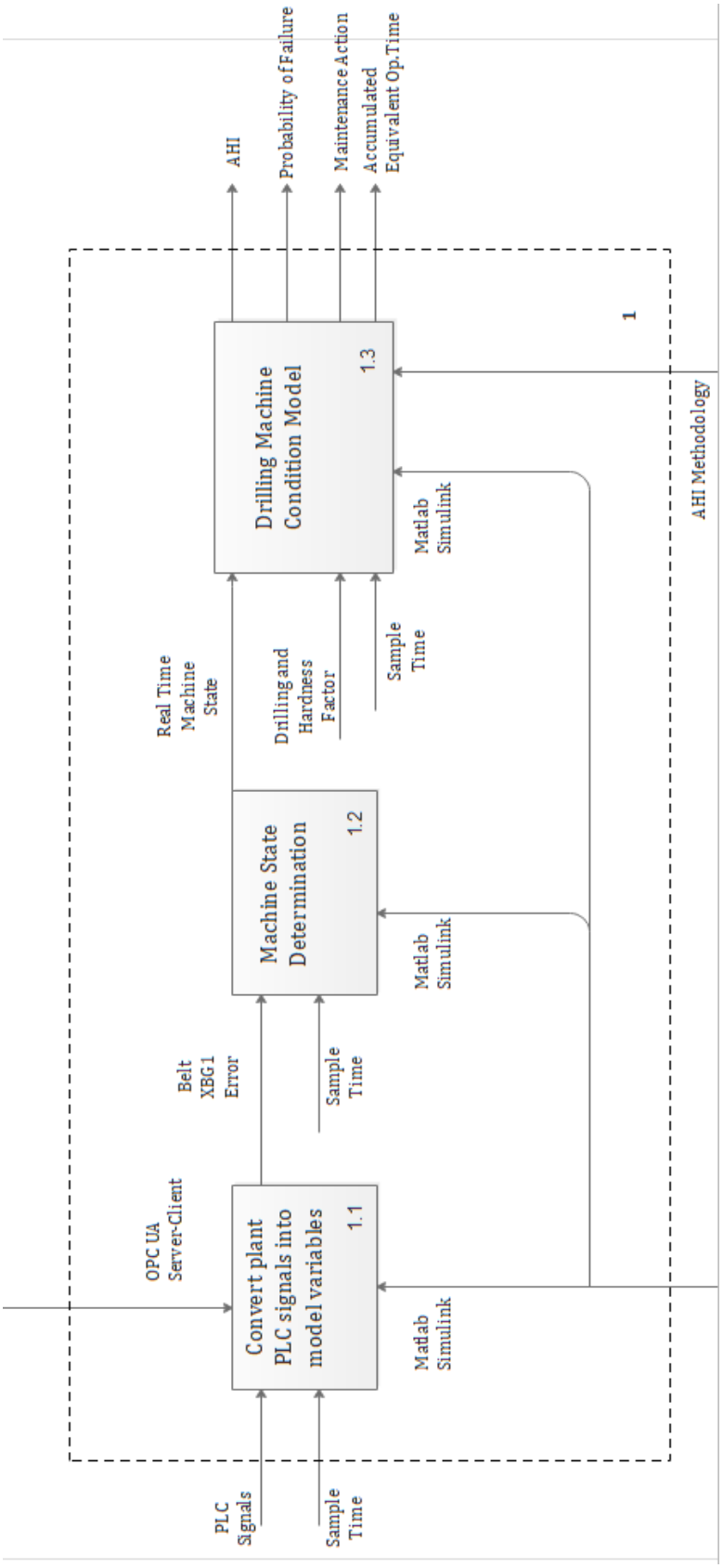


Figure 22: IDEF1 repesentation from the IDEFØ´s box 1

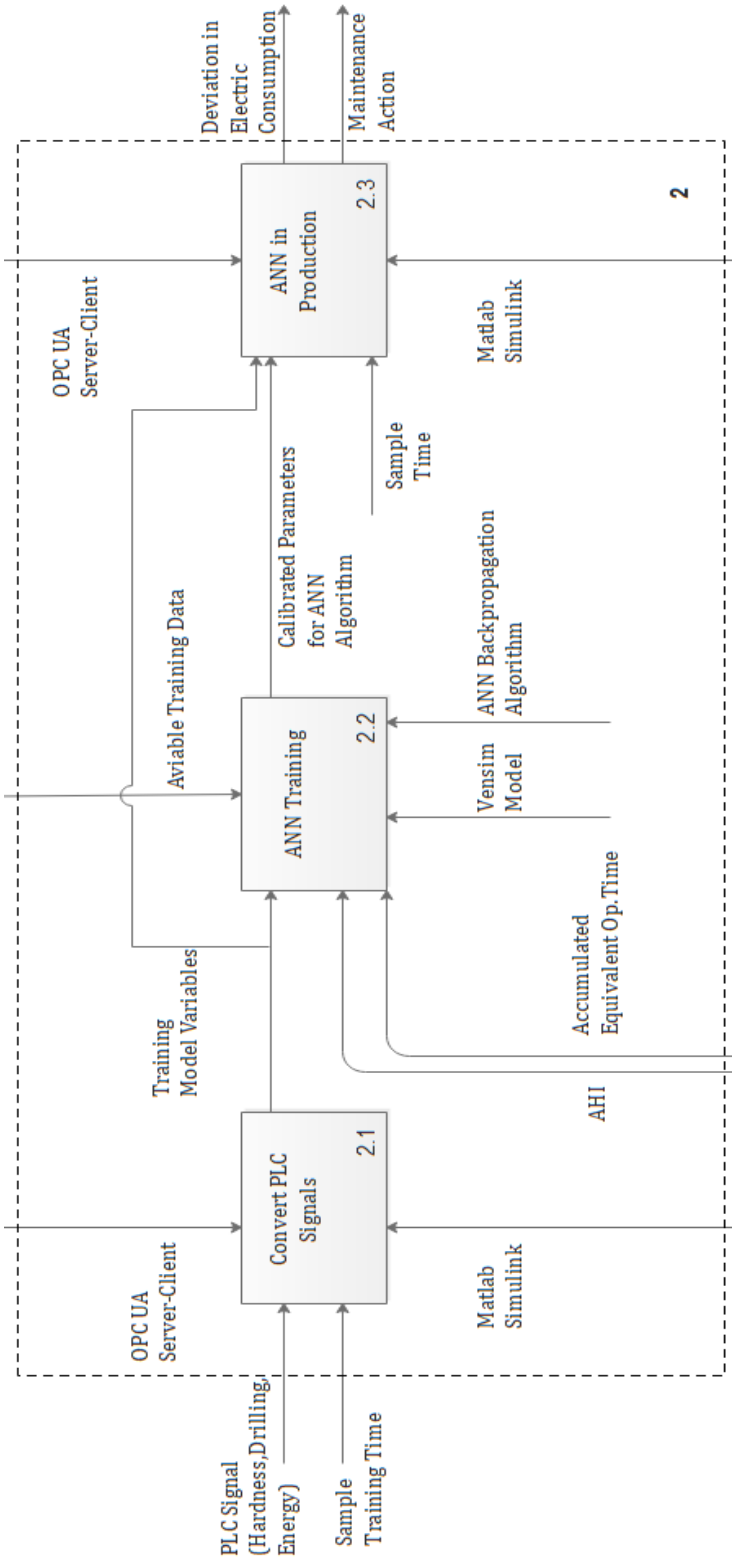


Figure 23: IDEF1 repesentation from the IDEFØ´s box 2

The last part of the IDEFØ box 2 (Figure 23), will be the implementation of the ANN in Matlab Simulink DT. And then make it work as another part of the multidisciplinary simulation model. Once the DT is completed with the ANN, the output that it could give is divided in two. The first one is the deviation in energy consumption of the ANN and the real values that will help to detect any anomaly in the asset. When detected that, a maintenance action will be executed.

It has to be said that the ANN hasn´t been implemented in Matlab, because the Hardness and Drilling factor of the machine are simulated, and the machine don´t actually degradates.

CHAPTER 5. MODELS DEVELOPMENT

5.1 Convert plant PLC signals into model variables

The chapter 4.4 refers to the *IDEF1, 1.1 box* (Figure 22). In the following chapter it is deeply explained how the conversion from a PLC signal to the Matlab variables (Belt, XBG1,Error) that are used to model the machine state.

The communication between Matlab/Simulink with PLC/sensors of the physical machine is realized with the OPC UA standard.

A level 2 Matlab S function and a simple logic Matlab function have been used to generate and transfer the input state signal. Since there is not a single sensor value that indicates the state of the machine, it is necessary first of all to find the sensors that could represent the states. There are more than 100 sensors of different types embedded in each CP module, unfortunately how the sensors are structured in the PLCs of the plant is unknown. The only way to find them was through trial with the help of UAexpert software.

5.2 Single Machine Model (Machine state determination)

Up to this point, the extraction of the signals from the PLC is being made. Then the *IDEF1 1.2 box* is developed (Figure 22). In the 4.5 section, the states of the machine are modelled, monitored at real time, and recorded in an historical state base.

The model has a virtual part and a corresponding machine-states subsystem which communicates with the real system in the Lab instead of by user-interface. Here this section only highlights the main difference between the model created with respect to industrial case and I4.0 Lab. According to the design of the system in I4.0 Lab, the module has four states that can be modelled:

- *Idle:* the conveyor in the module is moving, but no operation is performed.
- *Working:* the machine is performing operation.
- *Energy-saving mode:* the consumed energy is the amount needed only for keeping the module on.
- *Fault:* the machine is blocked due to abnormal behaviour and shows error message in the human-machine interface (HMI).

In this regard, the machine-states have been modelled as shown in Figure 24, they are labelled as 1, 2, 3, 4 respectively.

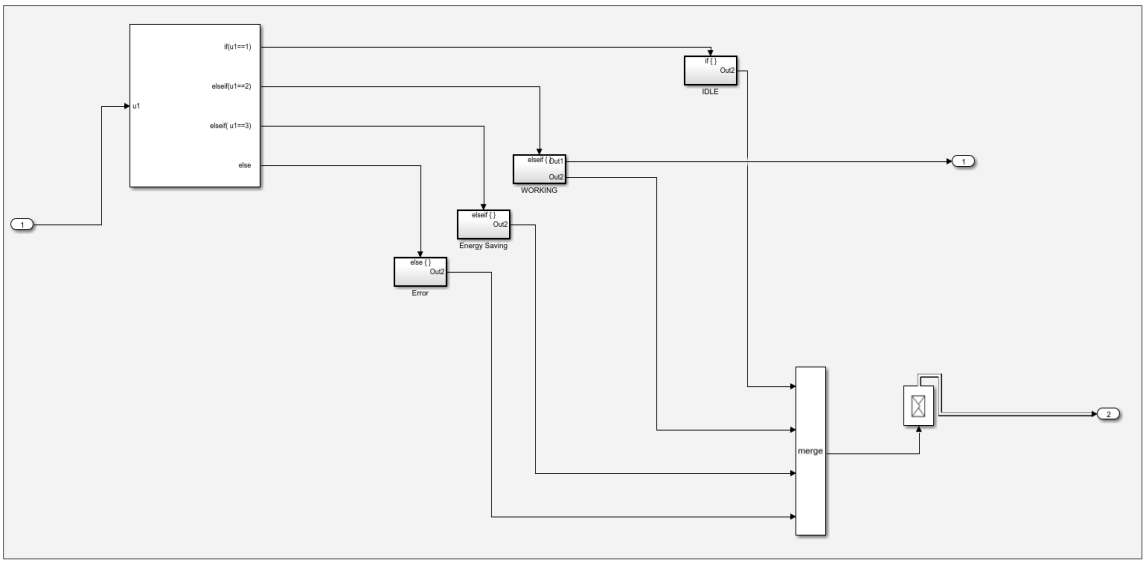


Figure 24: Subsystem mask that defines the state value.

A mask of the subsystem can allow the user to define the service time of the module. After the machine states have been identified and modelled, the connection to the real equipment needs to be implemented. Thus, as opposed to be controlled by the user-interface, this model communicates directly with the real system in the I4.0 Lab. At the end of the day, it turned out that three sensor values can be combined together to represent the machine states. They refer to the logical signal of the conveyor (xQA A1), the logical signal when the RFID is read (xBG1) and the integer signal of the fault (iRedCode).

The outputs of the logical variables of the level 2 S functions are 0 when the value is false and 1 when the value is true. When the part enters the drilling module, the signal xQA A1 becomes true (1), while all the other signals are 0, the machine is in idle state. When the part arrives to the stopper where the RFID on the carrier is read, signal xBG1 becomes true (1) and MES will tell the machine to perform the operation. When the machine starts working, the conveyor stops. When the work is done, the conveyor restarts, xBG1 turns to 0 and the part leaves the system, the machine is again in idle state. When the part leaves completely the module, the machine goes to energy saving mode. Regarding the fault state, the part will be blocked at the stopper, thus xBG1 and xQA A1 are both true, but iRedCode will return 4 as predefined through HMI (Table 2).

Conveyor (xQA	A1)	xBG1	iRedCode	State
1		0	0	IDLE
1		1	0	IDLE
0		1	0	WORKING
1		1	4	FAULT
1		0	4	FAULT
0		1	4	FAULT
	Others			ENERGY SAVING

Table 2: Signal combinations of Drilling Module

SIGNAL → Matlab Function Variable (INPUT)

Conveyor == belt
xBG1 == xBG1
iRedCode == error

These values are firstly read using the level 2 S function, and then a Matlab function is created to group the signals according to the combination shown in the Table 2 to output the machine state to be used by the virtual simulation model. The sampling time of the level 2 S function needs to be regulated in order to increase the simulation precision, in this case, 0.4 s has been chosen as the best one after performing several trials. And the simulation pace is set to be as close as the real clock time.

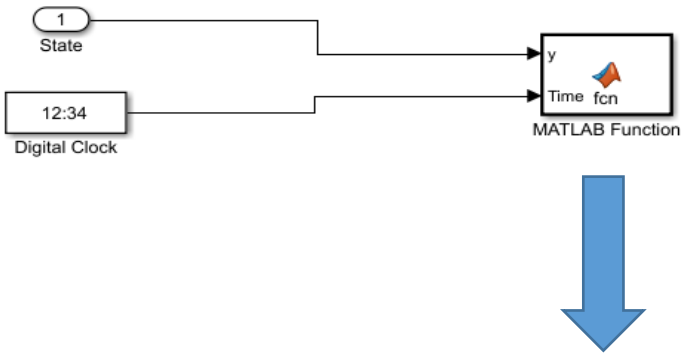
```
function y = fcn(belt, xBG1 ,error)
  if (belt == 1 & xBG1 == 0 & error == 0) | (belt == 1 & xBG1 == 1 & error == 0)
     y=1; %IDLE
     else if (belt == 0 & xBG1 == 1 & error == 0)
         y=2; % WORKING
         else if (belt == 1 & xBG1 ==1 & error == 4) | (belt == 0 & xBG1 ==1 & error == 4) | (belt == 1 & xBG1 ==0 &
             y=4; %ERROR
             else
             y=3; %ENERGY SAVING
             end
         end
end
```

Figure 25: Matlab function for Drilling Module

Matlab Function OUTPUTS

Y==1 → IDLE
Y==2 → WORKING
Y==3 → ENERGY SAVING
Y==4 → ERROR

Up to this point, the machine states in real time can be monitered. Firstly the signals needed are taken from the PLC as outputs with the Matlab Level-2-S-Function. Then the states can be modelled by these signals, and finally, the actual machine state in real time is known by using the Drilling Machine states block in the model as shown in Figure 25. Once this function of the DT is reached with success, the next objetive, was creating a data base that stored the machine states value through the simulation at real time. And the first step was making a new block in the model, which stored a function which receives the machine state value, and the real simulation time as inputs.




```
function  fcn(y, Time)
persistent fileID
if isempty(fileID)
fileID = fopen('Historical.txt','w');
fprintf(fileID,'%6s %12s\n\r','Time','Statevalue');
end
%Write a while loop for all the times from the start to the end of the
%simulation time
%while statevalue>0
fprintf(fileID,'\r\n%6.2f %12.8f\n\r',Time,y);
%fclose(fileID);

end
```

Figure 26: Matlab Historical state function

With the end of the block shown in Figure 26 , the single machine DT model that can monitor values at real time, and actually can also store the data from the simulation in real time in a txt file. Once that block model is is finished, it can be easily seen that a monitoring of the machine states evolution through real time simulation is reached, but then, the interesting point is knowing the real time that the machine has been in each state. In order to solve that question, it can be created a new block in the Matlab Simulink model that acts as an accumulator of the time in each state.

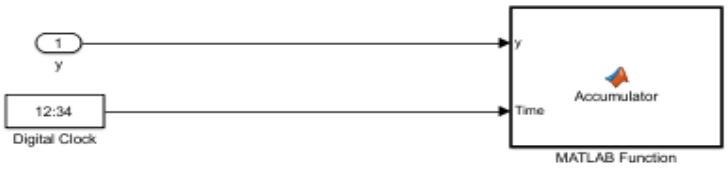


Figure 27: Matlab states time accumulation blocks

As the Figure 27 shows, the inputs are exactly the same, as the ones needed for the last function block above. But in this script, the accumulation of time is done and it is also printed in a txt, that shows four columns with the total amount of time in each state at each moment; and the total amount of time in each state throught the total simulation time will be shown in the last txt row as Figure 28 shows.

```
IdleTime  WorkingTime EnergySavingTime FailureTime
0.00000000   0.00000000   0.40000000   0.00000000
0.00000000   0.00000000   0.80000000   0.00000000
0.00000000   0.00000000   1.20000000   0.00000000
0.00000000   0.00000000   1.60000000   0.00000000
0.00000000   0.00000000   2.00000000   0.00000000
0.00000000   0.00000000   2.40000000   0.00000000
0.00000000   0.00000000   2.80000000   0.00000000
0.00000000   0.00000000   3.20000000   0.00000000
```

Figure 28: Time accumulator on each state

To sum up, at this point of the thesis, the DT has allowed to get and monitor the information from the machine that encompasses some of the most important aspects of the machine evolution through the time. In fact, these data and information will be a very useful allied to have a deeper understanding and knowledge of the machine lifecycle.

5.2.1 Single Machine model on Energy consumption

CPS has not only static information but also dynamic information, then the behaviour of the machine should be also integrated in the DT. With the increasing

emphasis on sustainability and environmental impact, industry production plant on one hand requires a big amount of energy, on the other hand it needs to conform with the regulations, thus energy consumption behaviour of the plant is essential to be monitored and then optimized [40] . To simulate and monitor the energy consumption in a production system, there is not a standard approach to implement the energy flow into the material flow simulation. Therefore, this work adopted the method proposed by Weinert [41]: EnergyBlocks methodology. The core of this methodology is based on the machine operation states. Each type of equipment has various operating states that exhibit different energy consumption patterns than can be identified in its power profile, and each operating state is defined as an EnergyBlock.

This method considers only the electrical energy. Machine energy consumption is physically continuous and hence the power profile of a machine is mathematically continuous. Continuous curves can be approximated as polynomials, each EnergyBlock represents a specific energy consumption behaviour. To formulate mathematically the energy consumption behaviour, two variables are considered. The required power P for one operation state within energy block m, and the duration of the corresponding operation state T. P can be approximated as the measured average value of the power profile within a specific state or adopted values from expert's experience and vendor specifications.

Regarding the energy consumption during the operation, in this work, only electrical energy is considered: that is the energy needed for the control devices (PLCs, Energy Boxes), for the conveyor transportation systems. The energy consumed by operations through the compressed air is not included. These energy consumption data with respect to each state should be obtained through the historical data of the energy database. The energy box displays only the instant value of the power consumption value.

5.3 Drilling Machine Condition Modelling

Regarding to the *IDEF1* skim (Figure 22), the development of the Drilling Machine Condition Modelling belongs to the *1.3 box* in the diagram. The machines in the Industry 4.0 Lab (Figure 31), don´t actually degradate so it is needed to recreate and simulate some factors that represent how the condition evolves through time.

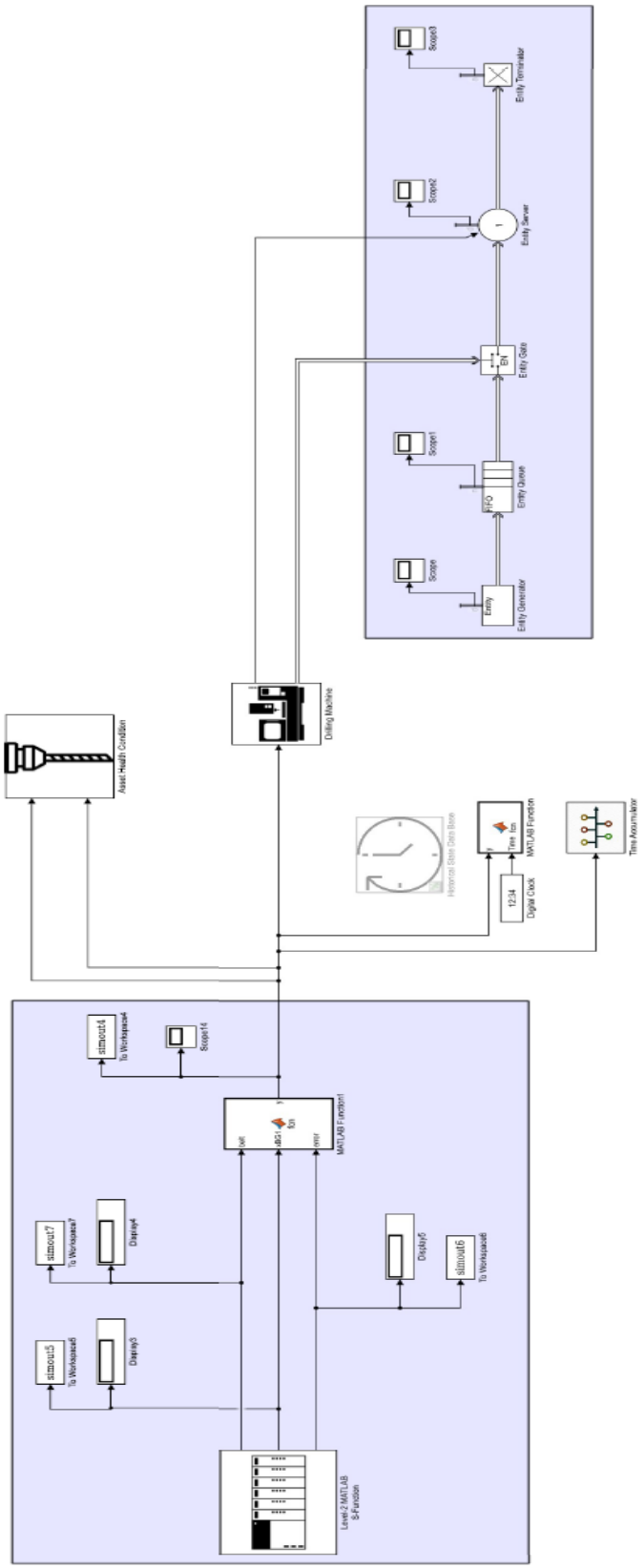


Figure 29: Matlab Simulink model for Drilling Machine Condition

In order to reach that, a simple simulation model (Figure 29) has been developed considering two simulated factors which are considered as the asset degradation factors, that are the Hardness factor of the material (Hardness of the material that the machine is drilling) and the Drill factor of the material (Numbrer of holes that the asset drills). The model takes as inputs these factors, as the real simulation time, and as a connection with the DT, the state of the machine at real time. Having the state of the machine is essential because the model considers that the asset only degradates when it is in working state. Both Idle and Energy Saving don´t degradate the asset condition.

At first is should be said that the model uses indicators from different levels depending on the deepness. To begin, the first level indicators have the same function as the time accumulator function from the Matlab Simulink DT model. It is done an accumulation of the asset´s time on each state, and actually on the same level, there is an indicator for the hardness factor and the drilling one. Until this point, the model has only done a monitoring of the real and simulated factors. From this monitoring comes the main model´s function. The second level of indicators is composed by three. There are two indicators that are more relationed to the monitoring aspect (Avaiability, Utilization), and another that could be more related with the asset working time (DH Equivalent Operating Time). The last one considers a value that comes frome the multiplication of both Drilling and Hardness factor. That equivalent time represents that the asset working time has to be modelled considering the caracteristics of the manufacture work, which means that the equivalent time is used to simulate the “real” degradation of the machine depending on the stress of the work done. In the same way, the equivalent operating time can be different to the time that the asset has been on the working state, just because the analisys takes in consideration how the condition of the asset changes from the maintenance view.

Once understood the DH Equivalent operating time, the next step is get in contact with the Accumulated Equivalent Operating Time. It has been mentioned in the previous page that the model considerates the “Working” state as the only one that affects the machine condition, and so on, the DH Eq OP time condition to be different from cero is the asset being working. Then the next level indicator, accumulates this equivalent time until a failure happends or a replacement of the asset is done (maintenance). So to sum up, this level three indicator accumulates the equivalent operating time while the asset is working and is not suffering a replacement of failure that will turn the indicator value cero (Figure 30).

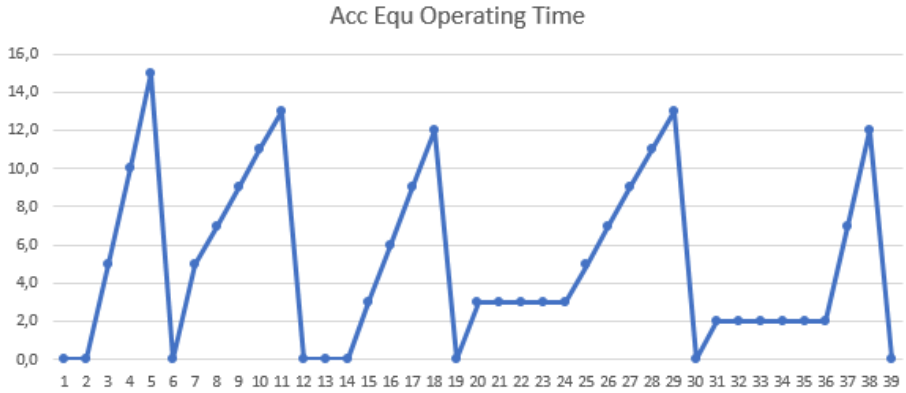


Figure 30: The Acc Equ Op Time over the simulation time

After the levels mentioned, the model goes deeper and gets to a fourth level. In this level an Ageing Rate of the asset is defined as a tool to modelate the Asset Health Index. The Ageing Rate is a constant built from the health index when the asset is new, at the end of the asset's useful life, and expected life. From the Ageing Rate and the Acc Equ Operating Time the model gets the value of the Asset Health Index (AHI), which is represented by an exponential function. The AHI represents how the health condition of the asset is evolving in time. It is also a representation of the degradation at real time, which is the main variable used to modelate the Probability of Failure variable (PoF). The PoF is a variable that represents the probability of failure that the asset has in each moment of time depending on the condition of the machine (AHI) [42][43][44].

Figure 31: Industry 4.0 Lab´s Drilling Machine

At this point of developing of the model, it can be focused or finished in two different ways. At first it can be done a Montecarlo simulation with a random number and a variable call Fail. And this first function can be developed just as a monitoring of the fail and the failure state. Then in the second path, there is an application which is

more connected with the DT model because it is more related with a preventive and predictive policy. By the way, it can be defined a variable called Failure probability threshold (FPT) that is a policy defined as a constant at the start of the model, representing the probability of failure that is accepted as a risk on the asset, before making the stop for maintenance. As a consecuence, the model needs a Boolean variable called Preventive in order to represent when the preventive maintenance stop happens. It can be easily understood that the Preventive value will be 1, when Pof is bigger than FPT, because it will mean that the probability of failure of the asset is bigger than the value considered as the asset´s limit failure risk. All mentioned above will mean that the model focused in this second waty does not consider the fail as something that will happend with a high probability. Just because there is defined a failure probability threshold, which will not let it happen if it is well defined depending on the asset and it´s operating time.

5.3.1. Model Notation and Equations

The notation us in the model is as follows:

- *Monitoring variables (inputs)*

  $ITime_t$*:* Boolean variable with value 1 if machine is in Idle state at time t

  $WTime_t$: Boolean variable with value 1 if machine is in Working state at time t

  $ESTime_t$: Boolean variable with value 1 if machine is in Energy Saving state at time t

  $FTime_t$: Boolean variable with value 1 if machine is in Fault state at time t

  HF_t: Hardness factor of the material at time t

  DF_t: Drill factor of the material, number of holes to drill at time t

- *Constants*

  AR: Ageing Rate of the asset

  LE: Life Expectancy of the system before the overhaul or replacement

  Hel: Healh index at expected life

  Hn: Health index when the asset is new

  Hel: Health index at the end of the asset's useful life

- *Dependent Variables*

  S_t: Machine State through the simulation real time

  PoF_t: Probability of failure at time t

F_t : Boolean variable to represent (value 1) a failure at time t

RN_t: Pseudo Random Number from a Uniform distribution between 0 and 1 at time t

$Preventive_t$: Boolean variable to represent (value 1) when the machine has to stop, to make the preventive maintenance.

FPT: Failure probability threshold

- *Level 1 indicator variables*

AIT_t: Accumulated Idle time at time t

AWT_t: Accumulated Working time at time t

$AEST_t$: Accumulated Energy Saving time at time t

AFT_t: Accumulated Failure time at time t

- *Level 2 indicator variables*

DH_t: Equivalent operating (considering Drilling and Hardness) time at time t

AV_t: Machine availability at time t

UT_t: Asset Utilization at time t

- *Level 3 indicator variables*

AOT_t: Accumulated asset operational time, at time t

AUT_t: Asset Utilization average through simulation real time

- *Level 4 indicator variables*

AHI_t: Asset Health Index at time t

With the above declared notation, the equations of the model can now be writen, and are as follows:

- *Input variables*

$$HF_t = \begin{cases} 1.5 - High\ hardness\ factor\ at\ time\ t \\ 1.2 - Medium\ hardness\ factor\ at\ time\ t \\ 1.0 - Low\ hardness\ factor\ at\ time\ t \end{cases} \quad \text{(Input)}$$

$$DF_t = \begin{cases} 1 - Two\ holes\ drilling\ at\ time\ t \\ 2 - Four\ holes\ drilling\ at\ time\ t \end{cases} \quad \text{(Input)}$$

$$ITime_t = \begin{cases} 1 & Idling\ at\ time\ t \\ 0 & any\ other\ case \end{cases} \quad (1)$$

$$WTime_t = \begin{cases} 1 & Working\ at\ time\ t \\ 0 & any\ other\ case \end{cases} \quad (2)$$

$$ESTime_t = \begin{cases} 1 & Energy\ Saving\ at\ time\ t \\ 0 & any\ other\ case \end{cases} \quad (3)$$

$$FTime_t = \begin{cases} 1 & Failure\ at\ time\ t \\ 0 & any\ other\ case \end{cases} \quad (4)$$

$$S_t = \begin{cases} 1, & if\ ITime_t = 1 \\ 2, & if\ WTime_t = 1 \\ 3, & if\ ESTime_t = 1 \\ 4, & if\ FTime_t = 1 \end{cases} \quad (5)$$

- *1st Level indicators*

$$\text{AIT}_t = \sum_{i=0}^{i=t} ITime_i \quad (6)$$

$$\text{AWT}_t = \sum_{i=0}^{i=t} WTime_i \quad (7)$$

$$\text{AEST}_t = \sum_{i=0}^{i=t} ESTime_i \quad (8)$$

$$\text{AFT}_t = \sum_{i=0}^{i=t} FTime_i \quad (9)$$

- *2nd Level indicators*

$$DH_t = \begin{cases} HF_t \times DF_t & S_t = 2 \\ 0 & e.o.c \end{cases} \quad (10)$$

$$AV_t = (\text{AIT}_t + \text{AWT}_t + \text{AEST}_t) / (\text{AIT}_t + \text{AWT}_t + \text{AEST}_t + \text{AFT}_t) \quad (11)$$

$$\text{UT}_t = (\text{AWT}_t) / (\text{AIT}_t + \text{AWT}_t + \text{AEST}_t) \quad (12)$$

- *3rd Level Indicators*

$$\text{AOT}_t = (AOT_{t-1} + DH_t) \times (1 - FTime_t) \quad (14)$$

$$\text{AOT}_0 = 0 \quad (15)$$

$$AUT_t = (\sum_0^t UT_t) / t \quad (16)$$

- *AHI (Asset Health Index)*

$$AR = \frac{\ln \frac{Hel}{Hn}}{LE} \quad (17)$$

$$Hel = 5{,}5 \quad (18)$$

$$Hn = 0{,}5 \quad (19)$$

$$LE = 10 \tag{20}$$

$$AHI_t = \mathrm{Hn} \times e^{AR \times AOT_t} \tag{21}$$

- *PoF Estimation*

$$PoF_t : \begin{cases} 0.01 & AHI_t \leq 5{,}5 \\ 0.0000359381 \times (e^{1.02337 \times AHI_t}) & 5{,}5 < AHI_t < 10 \\ 1 & AHI_t \geq 10 \end{cases} \tag{22}$$

$$F_t : \begin{cases} 1 & Random\ Number_t < PoF_t \\ 0 & a.o.c \end{cases} \tag{23}$$

$$Preventive_t : \begin{cases} 1 & PoF_t \geq FPT \\ 0 & a.o.c \end{cases} \tag{24}$$

5.3.2. Model Adaptation to Matlab Simulink

The model is developed above to have a better understand of the way it works, by the equations. Then it has to be adaptated to Matlab Simulink, in order to implement it in the DT model as another of the multidisciplinary simulations.

Firstly it has to bre created a function that generates a value for the simulated operational factors (Hardness and Drilling) when the state is working state. It is reached by creating a random uniform distributed number between 1 and 0. And depending on the number of values that the factor can get (3 for the hardness and 4 for the drilling one), create as many divisions on the distributions as values the factor can get (Figure 32).

```
if state == 2
    x1=rand;
    x2=rand;
if (x1<0.3)
    hardness=1;
    else if (x1<0.6)
        hardness=1.2;
        else
            hardness=1.5;
        end
end
```

Figure 32: Simulation of the values of the factors depending on a random distribution

If the values can be used as real inputs, because the simulation has been made, the model has all it needs to work. So the implementation is made as a new model block and to a large extent the AHI can be modelled and monitored for decision making.

5.4 Machine Artificial Neural Network (ANN)

Up to now, the equivalent operating hours are calculated to predict the moment from the risk of suffering a failure starts to increase. Basically the Drilling Machine Condition model provides a monitoring of the health of the asset by the AHI. It is made base to our information about the operating variables of the process.

Apart from that, we can also use other vigilance techniques focused on the process of appearance and propagation of the failure, based on deviations from the normal behaviour on a specific condition observed on the asset. For example, we know that the electrical consumption of the asset for a determined hardness, number of holes, and accumulated operating time is X (the machine has been trained and can calculate it), however it gives the value X + increase of X, then we can conclude that the failure process is developing itself, and how much time it will take.

This last idea could be reached by a Neural Network that is coordinated with the AHI (Asset health index). Writing down the times between failures, and by this way have a better model of the asset health, which can measure better the factors of the machine degradation. With the Neuronal Network, the failures can be detected before they physically happen. The inputs that can be used are the Drilling Factor, Hardness Factor, Accumulated Asset Operation Time, and the Asset Health Index for example. And as an output the Energy Consumption. These variables are provided to the ANN training model in the same way that they are given to the DT developed in the IDEFØ box 1. So the process of converting the PLC signals into model variables is totally the same. So the *IDEF1 2.1 box* (Figure 23), is considered to be developed in the same way. Then the *IDEF1 2.2 box* (Figure 23), gets the model converted variables, as well as the AHI and the other machine condition variables from the Drilling Machine Condition Modelling (*IDEF1 1.3 box,* Figure 22).

Once the ANN has been presented and explained, the work is going deeper on the equations and implementation.

5.4.1 A Continuous Time Dynamic Simulation Model for the ANN

The dynamic simulation model below, is the tool chosen to make the ANN training. It represents the *IDEF1 2.2 box* (Figure 23). Firstly the kind of simulation chosen is explained, and then the model and equations are shown. The model for training is been tested with simulated data, to prove it´s efficiency. But the model has not been tested with real data because the operational factors are not real, and due to that fact, the model has only been shown and tested with random simulated data.

5.4.2 The ANN Dynamic Simulation Model

In this Section a continuous simulation model of the previous ANN with backpropagation is presented. The dynamic simulation model that will be presented in this work can be characterised as follows:

- It is a non linear models because of the nature of the ANN.
- Difference equations will be used to formalise the models, i.e. future values of variables will be expressed as a function of the current and possibly past values.
- The time advance method will be time-stepped, i.e. time will be advanced in fixed time increments and the system state will be recalculated at each increment.
- The time that the physical system is modelled – physical time of the simulation – will depend on the purpose of the specific analysis to carry out. For instance of the specific prediction to be done in a machine, for a certain operating mode of the machine, etc.
- The modelling methodology to follow will be the one presented for System Dynamics. In this case special attention is paid to the use of the different models as decision support systems. The use of system dynamic tools such as the Stock and Flow diagram (SFD) is selected for this specific model.
- The simulation software tool used to build the models in this book is Vensim. Vensim® (Vensim is a registered trademark of Ventana Systems Inc.) provides high rigour for writing model equations. It adds features for tracing feedback loops. In addition "Causes Tree" and "Uses Tree" features help in debugging the model. Vensim also provides very powerful tools for multiparametric simulation results optimisation which allows the analyst to validate results and model structure as well as to determine most convenient policy options by parametrising these policies. This work benefits, on several occasions, from the advantage of Vensim given by the incorporation of a powerful optimiser based on a modified Powell method algorithm. This feature produces very fast convergence of the direct search technique when optimising solutions and without the requirement of gradient assessment in the different iterations. Having said this, it is important to remember that the mathematical formulation of the models in this Chapter does not take into consideration the software used, *i.e.* Vensim code is only included in an Annex, so any can build the model regardless of the software tool preferred.
- Regarding model validation, the model will follow serious reality checks and validation procedures.

In the following Sections, first the the Stock and Flow Diagram (SFD) will be shown, and then the notation of the variables included in the model will be presented. Then

the simulation model formal equations will be listed. Finally some results and graphs of the model will be plotted and also the Vensim simulation model equations will be listed, in an Annex.

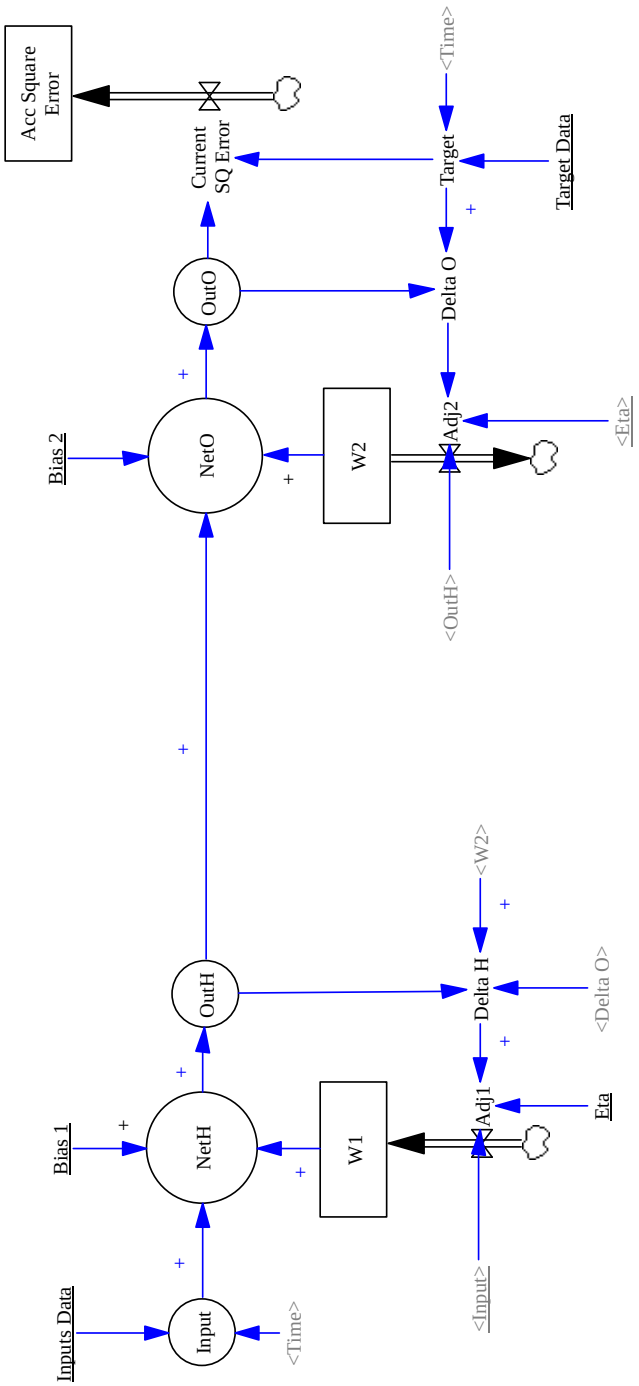


Figure 33: Stock and flow diagram of the ANN simulation model for backpropagation

Our example ANN dynamic simulation model SFD is presented in Figure 33. In this Figure the boxes, stock variables, are the weights in each layer ($w1_t$ and $w2_t$), and the accumulated square error during training (Acc Square $Error_t$). This last stock

variable has been added to compare simulation and measure the performance of the network under diverse scenarios. The weights may change according to the value that rates variable take over time. Rates of adjustments of the weights are $Adj1_t$ and $Adj2_t$, and they include the variation of the weights in each iteration that is defined by the Delta Rule in each layer.

The weights will be changing over time when the network is learning, and will remain constant once the network training ends. So Figure 33 represents the ANN simulation model for the process of backpropagation. Some of the Figure 5 variables are "shadow" variables, in order to avoid the influence arrows crossing or overlapping in the diagram. Table, constants or parameters are normally underline in this type of diagrams. Some of the auxiliary variables are placed inside a circle to represent the neurons (this would be no formal requirement in the SFD).

5.4.4 Notation of the variables

The notation that we will use together with the indication of the typology of the variable used, in Figure 5, are as follows:

- *Tables, Constants & Parameters*

  $Bias1 = Bias\ in\ the\ total\ input\ of\ the\ hidden\ layer\ neurons$

  $Bias2 = Bias\ in\ the\ total\ input\ of\ the\ output\ neuron$

  $Input\ (i)\ data = Lookup\ Table\ with\ normalized\ Input\ i\ data\ over\ time$

  $Target\ data = Lookup\ Table\ with\ normalized\ target\ data\ over\ time$

  $Eta = Learning\ coefficient$

- *Auxiliary variables*

  $Input(i)_t = Normalized\ Input\ i\ of\ the\ NN\ at\ interval\ t$

  $Target_t = Normalized\ Ideal\ output\ of\ the\ NN\ at\ interval\ t$

  $NetH(j)_t = Net\ input\ for\ the\ hidden\ layer\ neuron\ j\ at\ interval\ t$

  $OutH(j)_t = Output\ of\ the\ hidden\ layer\ neuron\ j\ at\ interval\ t$

  $NetO_t = Net\ input\ for\ the\ output\ layer\ neuron\ at\ interval\ t$

  $OutO_t = Output\ of\ the\ output\ layer\ neuron\ at\ interval\ t$

$Delta\ O_t = Delta\ function\ (\delta o_t)\ of\ the\ output\ at\ interval\ t$

$Delta\ H(j)_t = Delta\ function\ of\ the\ h\ layer\ (\delta H1(j)_{t)}\ neuron\ j\ at\ interval\ t$

- *Stock variables*

  $w1(i,j)_t = Weights\ to\ obtain\ the\ hidden\ layer\ neurons\ inputs\ at\ time\ t$

  $w2(j)_t = Weights\ to\ obtain\ the\ output\ layer\ neurons\ inputs\ at\ time\ t$

  $Acc\ Square\ Error_t = Accumulated\ Square\ Error\ at\ time\ t$

- *Flow Variables:*

  $Adj1(i,j)_t = Adjustment\ of\ weights\ w1(i,j)\ in\ time\ interval\ t$

  $Adj2(j)_t = Adjustment\ of\ weights\ w2(j)\ in\ time\ interval\ t$

  $Current\ SQ\ Error_t = Error\ in\ time\ interval\ t$

5.4.5 Equations

The equation for each one of the above declared variables are now presented::

- *Tables, Constants & Parameters*

  $Bias1 = B_1, \quad data$ (1)

  $Bias2 = B_2, \quad data$ (2)

  $Input\ (i)\ data = finput(x) \quad data\ table$ (3)

  $Target\ data = ftarget(y) \quad data\ table$ (4)

  $\eta = \eta o, \quad data$ (5)

- *Auxiliary variables*

  $Input(i)_t = finput(t)$ (6)

  $Target_t = ftarget(t)$ (7)

  $NetH(j)_t = \sum_{i=1}^{i=n} Input(i)_t \cdot w1(i,j)_t + Bias1$ (8)

  $OutH(j)_t = \frac{1}{1+e^{-NetH(j)t}}$ (9)

  $NetO_t = \sum_{j=1}^{j=m} OutH(j)_t \cdot w2(j)_t + Bias2$ (10)

$$OutO_t = \frac{1}{1+e^{-NetO_t}} \quad (11)$$

$$Delta\ O_t = (OutO_t - Target_t) \cdot OutO_t \cdot (1 - OutO_t) \quad (12)$$

$$Delta\ H(j)_t = Delta\ O_t \cdot w2(j)_t \cdot OutH(j)_t \cdot (1 - OutH(j)_t) \quad (13)$$

- *Stock variables*

$$w1(i,j)_t = w1(i,j)_{t-1} - Adj1(i,j)_t \quad (14)$$

$$w1(i,j)_{to} = w1o(i,j), \qquad Initial\ condition \quad (15)$$

$$w2(j)_t = w2(j)_{t-1} - Adj2(j)_t \quad (16)$$

$$w2(j)_{to} = w2o(j), \qquad Initial\ condition \quad (17)$$

$$Acc\ Square\ Error_t = Acc\ Square\ Error_{t-1} + Current\ SQ\ Error_t \quad (18)$$

$$Acc\ Square\ Error_{to} = 0, \qquad \text{Initial condition} \quad (19)$$

- *Flow Variables:*

$$Adj1(i,j)_t = -Eta \cdot\ Delta\ H(j)_t \cdot Input(i)_t \quad (20)$$

$$Adj2(j)_t = -Eta \cdot Delta\ O_t \cdot OutH(j)_t \quad (21)$$

$$Current\ SQ\ Error_t = \frac{1}{2} \cdot (Target_t - OutO_t)^2 \quad (22)$$

5.5. ANN Training

5.5.1. Calibration of the model parameters for ANN Training

Understanding ANN Training process dynamics is important. When the number of hidden layers grow, or the number of neurons per layer increases, the training process can be long and the opportunity to use the network appropriately could be jeopardized.

The training process implies feedback mechanisms that get activated to diminish the error. These mechanisms are robust, involving different goal seeking feedback loops with non-lineal behavior. However, non-linearities may result in a very slow movement towards the optimum when training the network, and the initial value for the set of parameters to be determined could result of vital importance to train properly the network with a reasonable amount of data.

In the following paragraphs and figures, as an example, existing feedback loops in the system dynamics model developed are presented, with the idea to identify completely those mechanisms generating training dynamics.

In Figure 34 the first more direct negative feedback loops are identified. The red loop (named 1) adjust the weights w2(j) to diminish the error $Delta\ O_t$, notice how the gain of the loop is conditioned by the value of Eta. The closer the value to 1 of Eta, i.e. the higher the learning coefficient, the larger the feedback effect to correct the error. This could be counterproductive when reaching points closer to the optimum. Bias 2 is also important in the sense that, Bias 2 may provide a fast initial approximation to the expected range of better w2(j) values, however it seems that final approximation mechanism to the optimum w2(j) values will be controlled by Eta. The slower Eta the lesser oscillation around the optimum, but the slower speed of approximation too.

The blue loop (named 2) adjust the weights w1(i,j) to diminish the error $Delta\ H_t$, notice how again the gain of the loop is conditioned by the value of Eta, but $Delta\ O_t$, and w2(j) may also impact on the loop gain. When the red loop is closer to the desired value of w2(j), then the error will be smaller and the gain of the blue loop feedback loop will also be lower. Bias 1 has in the blue loop a similar effect that Bias 2 has in the red loop.

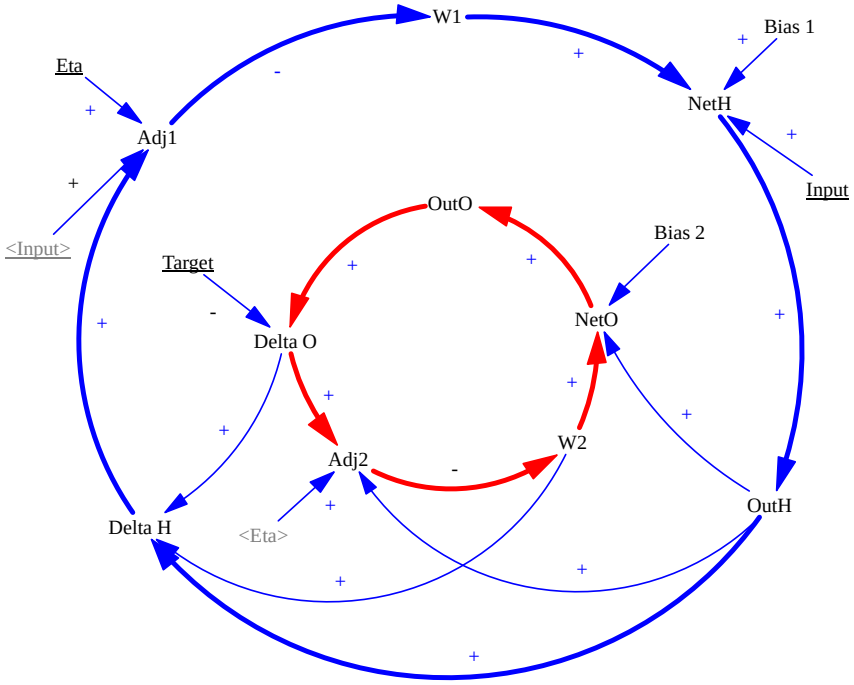


Figure 34. Negative feedback loops 1 (Red) and 2 (Blue)

Red loop dynamics impact on the blue loop ones. In Figure 35 a third negative balancing feedback loop is identified. This loop shows how we adjust w1(i,j) depending on the error of the output of the ANN. It is interesting to see how the weights adjustment, in the two levels, are interrelated.

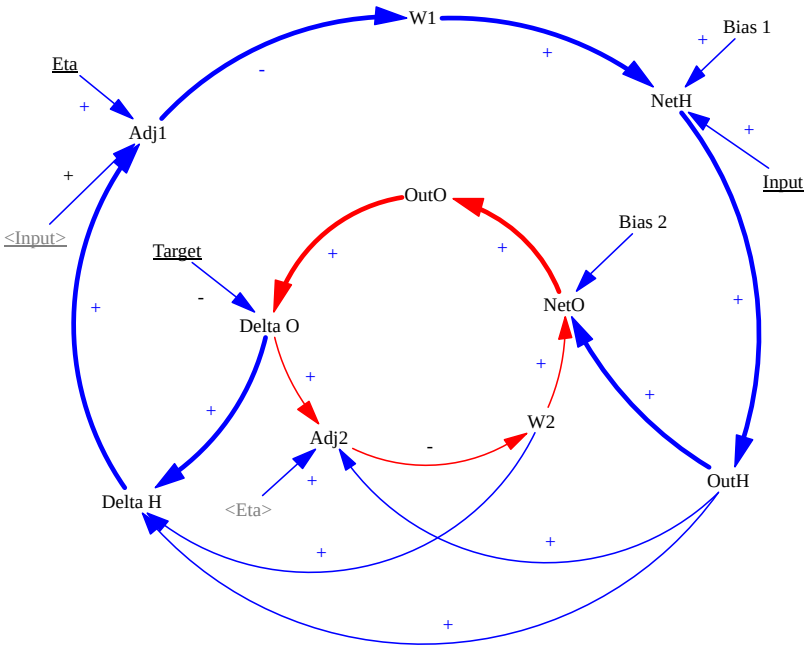


Figure 35. Negative feedback loop 3 (Red and Blue thick line)

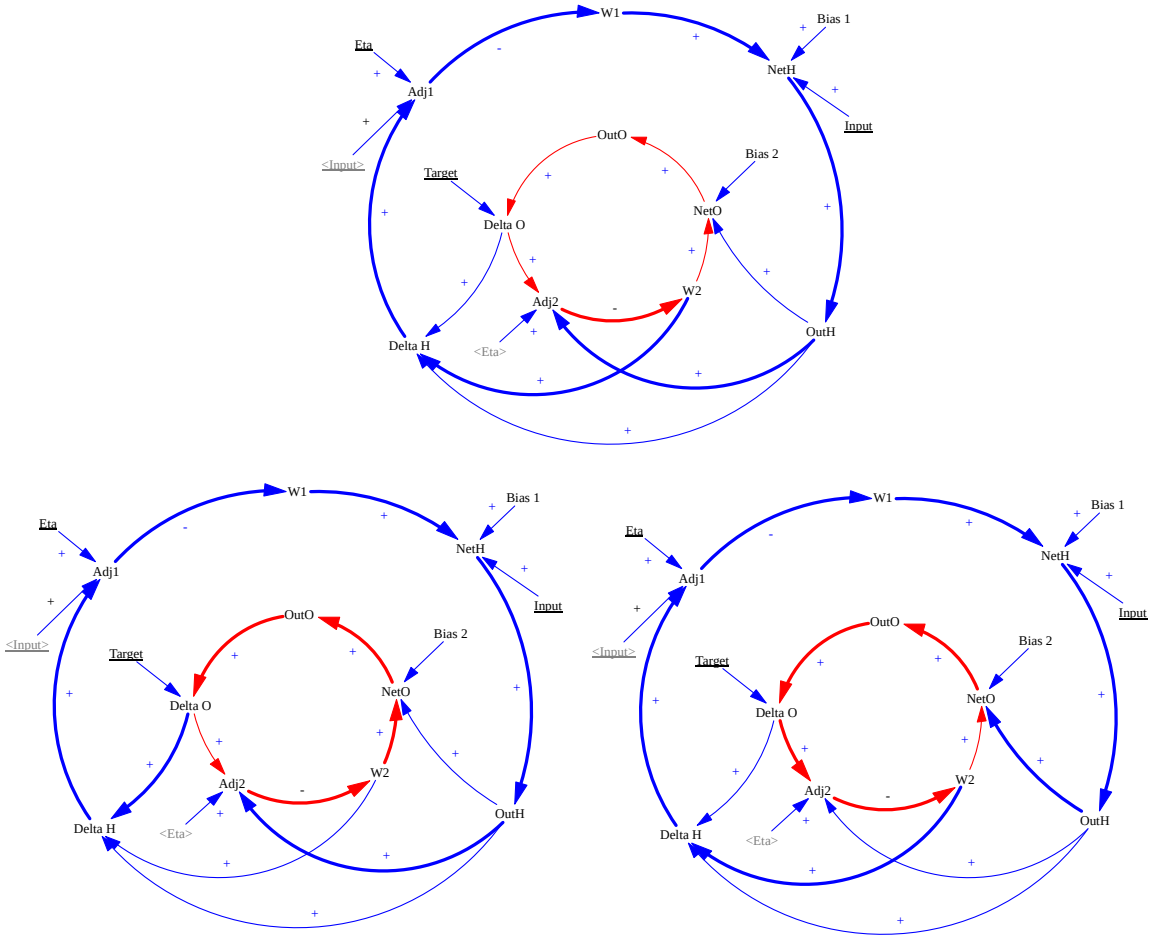


Figure 36. Positive reinforcing feedback loops 4, 5 and 6 (Red and Blue thick lines)

Figure 36 shows the three reinforcing loops existing in the ANN training model. These loops force the training model to replicate the effect of increasing or decreasing the adjustment of one set of weights $Adj1_t$ in the second set $Adj2_t$, and vice versa.

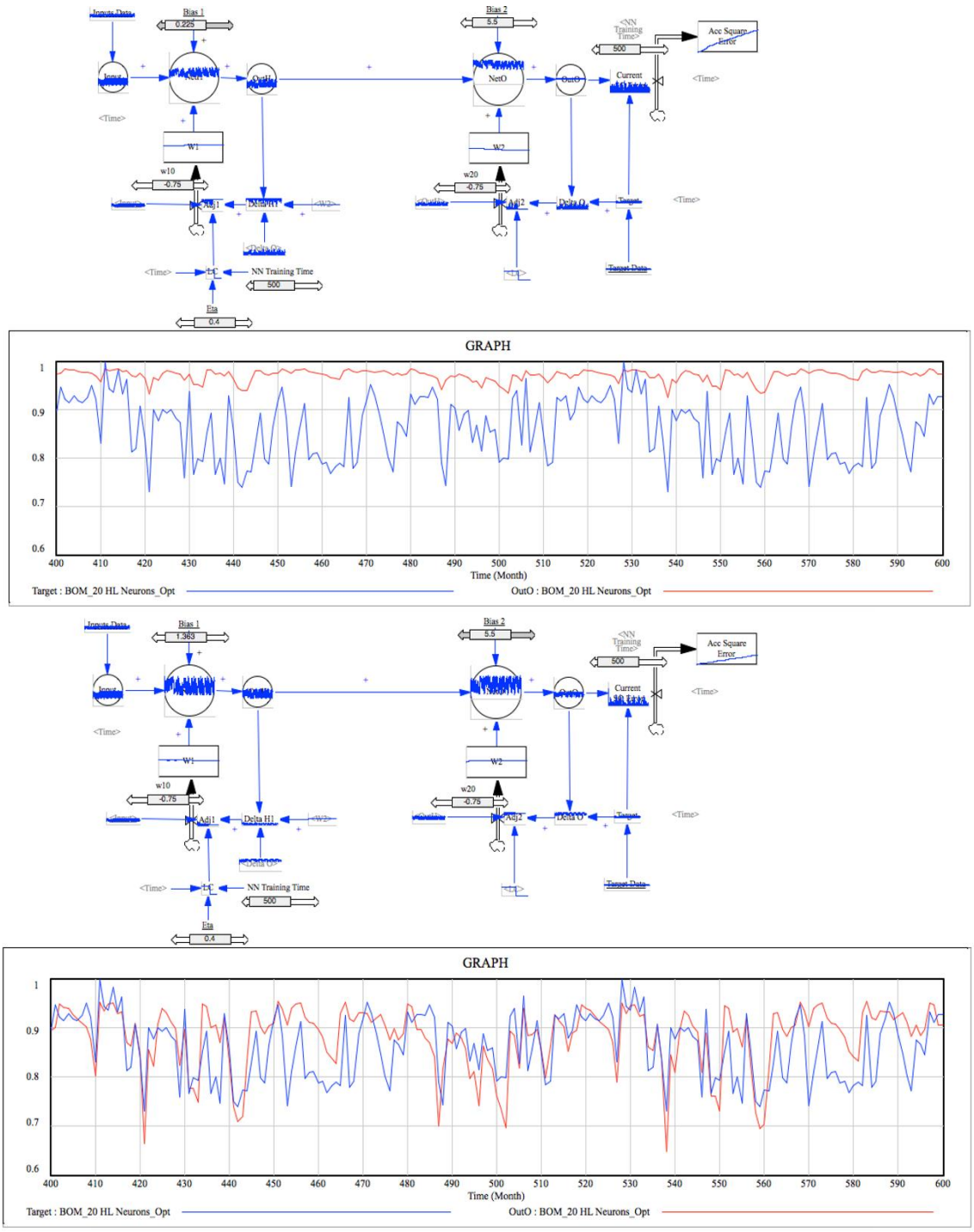


Figure 37. Effect of changes in Bias 1 parameter during ANN fitting

Some final learnings of this feedback loops analysis could be stated as follows:

- Selecting initial values for Bias 1 and Bias 2 parameters seems to be crucial to save time, to avoid training time in less precise, gross initial fitting, specially when eta is selected low. In Figure 37 and 38, some examples illustrating the effects of Bias selection considering the same initial weights values departing points are presented. By playing with Synthesim using Vensim software we could appreciate that it is important both values to be compensated for a reasonable fit. So these would be parameters whose values are very important to optimize for a fast convergence to the ANN weights to the proper target fit.

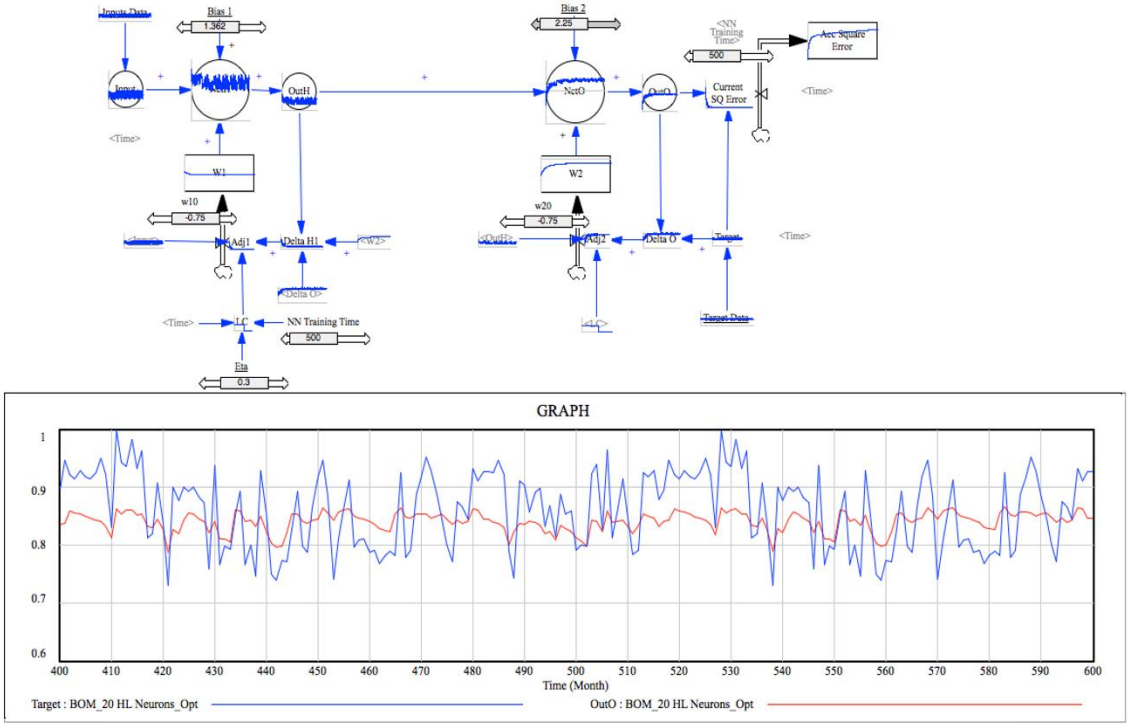


Figure 38. Effect of changes in Bias 2 parameter during ANN fitting

- Selecting initial values for Bias 1 and Bias 2 parameters seems to be also crucial to reach a better fit. Once Bias 1 and 2 are fixed the best possible fit is conditioned.
- Selecting proper Eta values depend on the right choice of weight initial values. When initial w1(i,j) and w2(j) values are randomly selected, Eta value becomes critical. When initial search points for this weights can be optimized, Eta impact on training time will be much less important. See Figure 39 with far from optimum initial values of both w1(i,j) and w2(j).
- Of course, we may expect that all previous problems related to training mechanism and dynamics are amplified when the number of neurons or layers of the ANN increase.

To overcome all these training dynamic issues we can rely on tools provided by Vensim for optimization and calibration of model parameters.

Vensim uses the direct-search method that does not evaluate the gradient (Powell Modified Method), to calibrate model parameters. In our case it is important to calibrate Bias 1, Bias 2 and Eta (learning coefficient) parameters in order that the ANN offers its best possible fit as soon as possible.

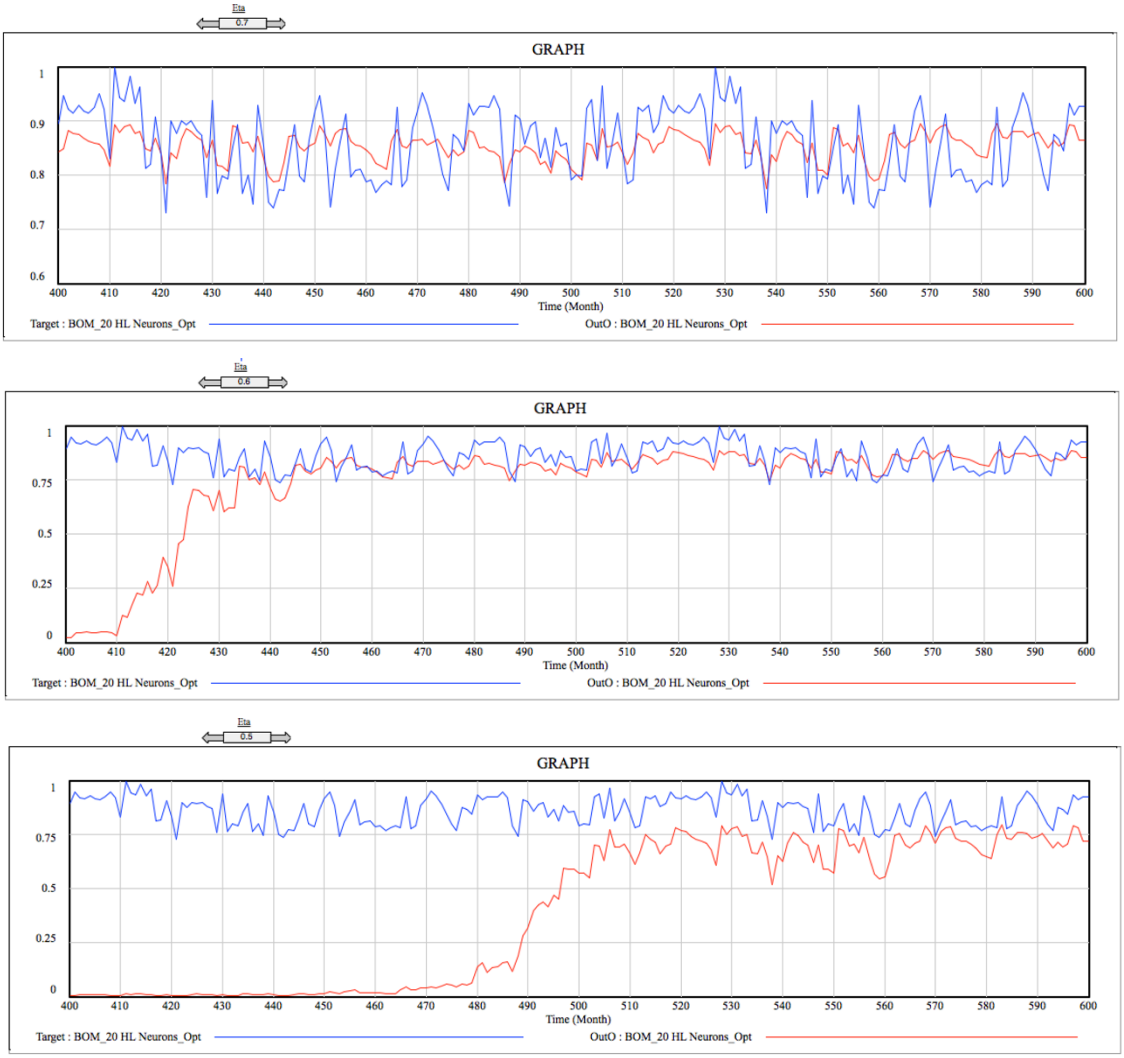


Figure 39. Effect of changes in Eta parameter during ANN fitting

The Powell method has been found to be very suitable for the analysis of dynamics of complex nonlinear control systems. The Powell method (Powell, 1964), is well known to have an ultimate fast convergence among direct-search methods. The basic idea behind Powell's method is to break the N dimensional minimization down into N separate one-dimensional (1D) minimization problems. Then, for each 1D problem a binary search is implemented to find the local minimum within a given range. Furthermore, on subsequent iterations, an estimate is made of the best directions to use for the 1D searches (In the Powell method, at most m iterations,

where m is the number of parameters to be estimated, yields the optimal solution to the problem with cost function of quadratic form, if the directions of m-dimensional vectors are linearly independent at every iteration step). Some problems, however, are not always assured of optimal solutions because the direction vectors are not always linearly independent. To overcome this difficulty, the method was revised (Powell,1968) by introducing new criteria for the formation of linearly independent direction vectors. This revised method, is called "The Modified Powell Method", which is the one used in this work. At present, there is a wide applicability of this method (see for instance the optimization module of VENSIM 5.1), and at the same time it is used in some new developments of hybrid numerical optimization techniques incorporating genetic algorithm into that method (Okamotoa et al. 1998).

For example, in our example with 8 neurons, if we introduce the initial point of search:

Bias 1 = 0
Bias 2 = 1
Eta = 1

An the result of the Power optimizer is: Payoff (-31.6176) realized at multiple parameter values after Simulations = 876, Optimizations = 36, Pass = 3. Select for instance:

Bias 1 = 1
Bias 2 = 0.416667
Eta = 1

Then we can select these values as the model parameters. Calibration can be done for each network configuration selection (i.e. number of HL neurons), and each time the target functions and inputs change.

5.5.2. Models Results and Validation

As sample model results we do present, in Figure 40, the evolution of the current and accumulated error over time when using 5 to 8 neurons in the hidden layer. This is a three inputs and the reader can appreciate a fast convergence to acceptable prediction of the ANN5 but then less precision than the ANN8, which tales longer time to adapt and therefore more accumulated error, but then seems to be more precise over time.

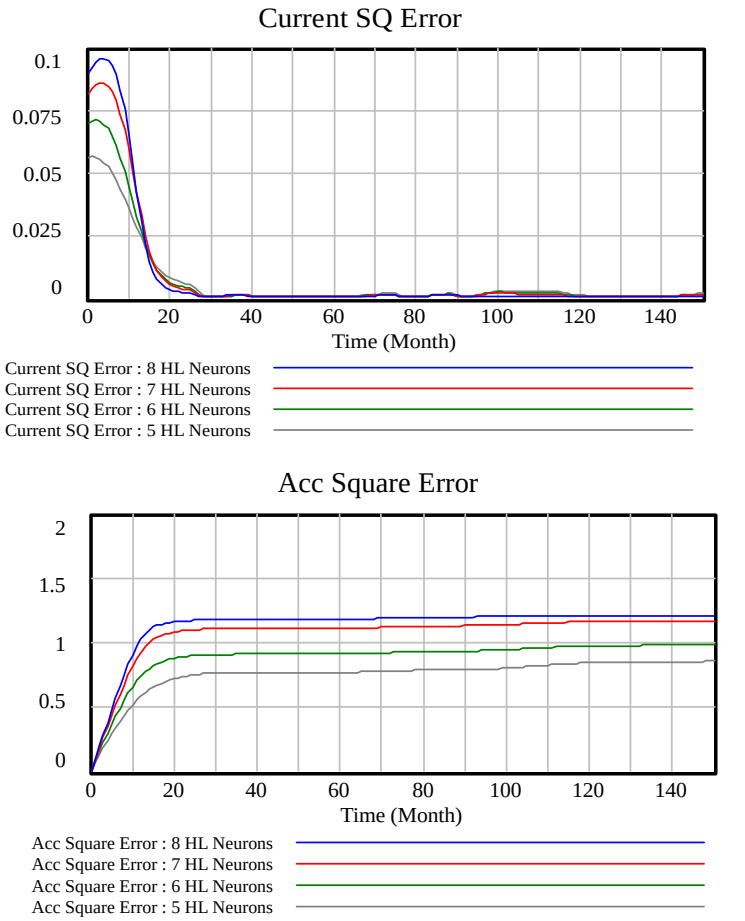


Figure 40. Current and Accumulated Error over time when using 5 to 8 neurons in the HL

In Figure 41, the evolution of the ideal versus current prediction when the network is trained is presented, when using 5 to 8 neurons in the hidden layer. The reader can appreciate the convergence to acceptable predictions when departing from W1(i,j)t0 = 0,5 and W2(j)t0 = 0,5.

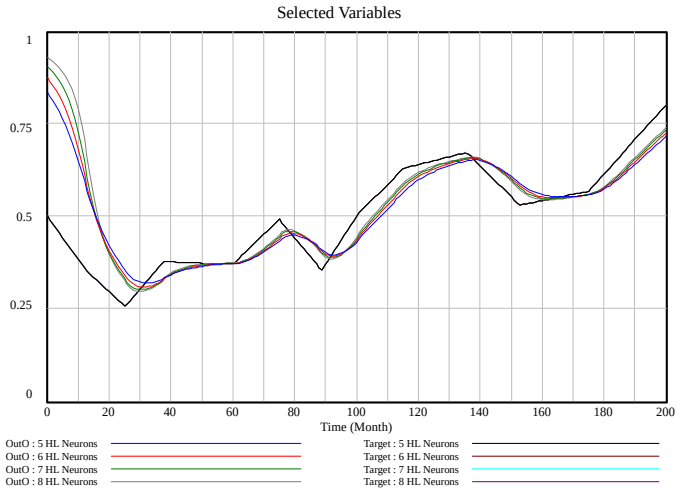


Figure 41. Target and Prediction of the ANN with 5 to 8 neurons in the hidden layer

CHAPTER 6. RESULTS DISCUSSION

As shown in the previous chapters, multidisciplinary models have been developed to have a complete vision work of the full concept of the DT. Once shown the methodology and implementation, it is interesting to have a discussion on the results of each part of the work development. In this chapter, the results of each model on chapter 5, are shown and explained.

At first, in order to perform the simulation, it is obliged to connect to the access point or the wireless of I4.0 Lab. This network grants access to all the facilities in the system: application modules, PLCs, energy boxes and so no and it is the basis of the OPC UA server/client connection, the signals from the PLC are taken and converted successfully by the Matlab Level 2 S-function. This can be considered the first result. The simulation on the Matlab Simulink DT gives at first the value of the state through a txt file in real simulation time (Figure 44). As well, the model provides an historical accumulator of the time in each asset state (Figure 44). These are the most important results, and apart from that, the modules as the state variable or the entity terminator (pieces ended in the simulation) provide a graphic that represents the values of the module variables in simulation, at the end of it (Figures 42, 43).

All mentioned above can be easily seen in an example run. To start the user defines and launches the production order in the MES. For example, which product do you want to process? How many pieces do you want to produce? And when do you want to start the production? All these questions can be clarified in the MES. In order to create a new product, two steps are needed:

1. *Create new work plan*: it is necessary to define first of all the work plan that is the process sequence to realize the product. Since in this case, there is only one machine, a single drilling operation is needed to be defined.
2. *Create new production part:* After the work plan has been created, production part needs to be added inserting the created work plan.

On this example run it can be seen that, as the work and production plan define, the entities entered the system one after another, the module did not have time to turn to energy saving state. Therefore, the states graph shows a clear and smooth pattern of “idle-working-idle” (Figure 42). No energy consumption data for the fault state is expected. By comparing with other runs it can be identified that with a tight scheduling, the idle time of the machine can be reduced and results in a lower consumption of energy.

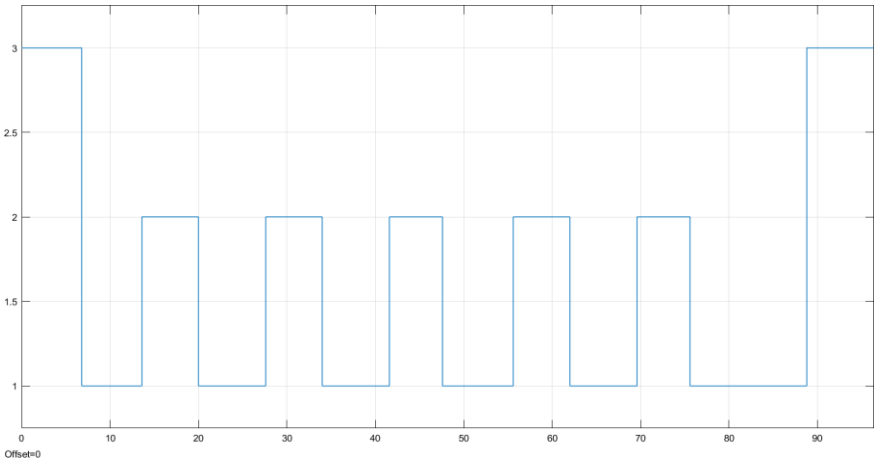


Figure 42: Machine states on an example run

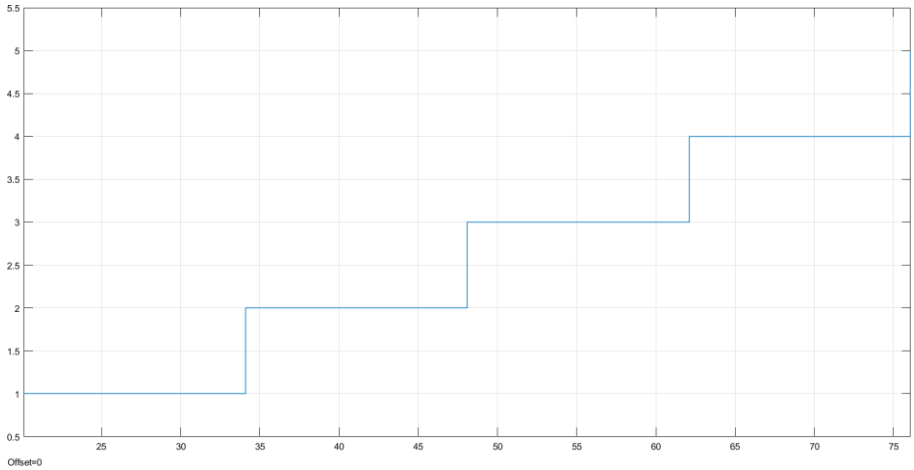


Figure 43: Entity Terminator of an example run

Time	Statevalue
0.00	3.00000000
0.40	3.00000000
0.80	3.00000000
1.20	3.00000000
1.60	3.00000000
2.00	3.00000000
2.40	3.00000000
2.80	3.00000000
3.20	3.00000000
3.60	3.00000000
4.00	3.00000000
4.40	3.00000000
4.80	3.00000000
5.20	3.00000000
5.60	3.00000000
6.00	3.00000000
6.40	3.00000000

IdleTime	WorkingTime	EnergySavingTime	FailureTime
0.00000000	0.00000000	0.40000000	0.00000000
0.00000000	0.00000000	0.80000000	0.00000000
0.00000000	0.00000000	1.20000000	0.00000000
0.00000000	0.00000000	1.60000000	0.00000000
0.00000000	0.00000000	2.00000000	0.00000000
0.00000000	0.00000000	2.40000000	0.00000000
0.00000000	0.00000000	2.80000000	0.00000000
0.00000000	0.00000000	3.20000000	0.00000000
0.00000000	0.00000000	3.60000000	0.00000000
0.00000000	0.00000000	4.00000000	0.00000000
0.00000000	0.00000000	4.40000000	0.00000000
0.00000000	0.00000000	4.80000000	0.00000000
0.00000000	0.00000000	5.20000000	0.00000000
0.00000000	0.00000000	5.60000000	0.00000000
0.00000000	0.00000000	6.00000000	0.00000000
0.00000000	0.00000000	6.40000000	0.00000000

Figure 44: Real time state monitoring, and real time on state accumulator druing the simulation.

Regarding the energy consumption during the simulation, only electrical energy is considered: that is the energy needed for the control devices (PLCs, Energy Boxes), for the conveyor transportation systems. In this end, in an energy point of view, the fault state is the same as idle state and the working state is the same as energy saving state since the conveyor stops when the module is doing operation (Table 3). These energy consumption data with respect to each state should be obtained through the historical data of the energy database. The energy box displays only the instant value of the power consumption value. Thus, by tracking by eyes the values on the display of the energy boxes during the production cycle, the following power consumption data are obtained. Then an accumulation of the energy consumption in each state depending on the real simulation time can be also done.

State	Power (w)
Idle	58
Working	43
Fault	58
Energy Saving	43

Table 3: Power consumption profile on each state on sample time

Once finished the monitoring aspect of the DT, and after the implementation of the AHI model on the DT; the new results that can also be provided by the model are the ones that we use as inputs for the ANN model, knowing of course that the operational factors are simulated. Up to this point, the Drilling and Hardness factors, the AHI, the accumulated operational, and the probability of failure are given by the model at real time by a txt file or could be implemented a mask that show all the information. In the same run done before, the values of al the variables above explained in the Drilling Machine Condition Model are provided as shown in Figure 45.

In the Drilling Machine Condition Model, it is not defined a value for the Failure Policy, and the maintenance action is also not implemented. These variables and actions are not implemented because the machine factors are simulated. If there were real, in a further work it should be developed a connexion back with the PLC to stop the machine for the maintenance action.

Time	AssetHealth
0.00	0.50000000
0.40	0.50000000
0.80	0.50000000
1.20	0.50000000
1.60	0.50000000
2.00	0.50000000
2.40	0.50000000
2.80	0.50000000
3.20	0.50000000
3.60	0.50000000
4.00	0.50000000
4.40	0.50000000
4.80	0.50000000
5.20	0.50000000
5.60	0.50000000
6.00	0.50000000
6.40	0.50000000

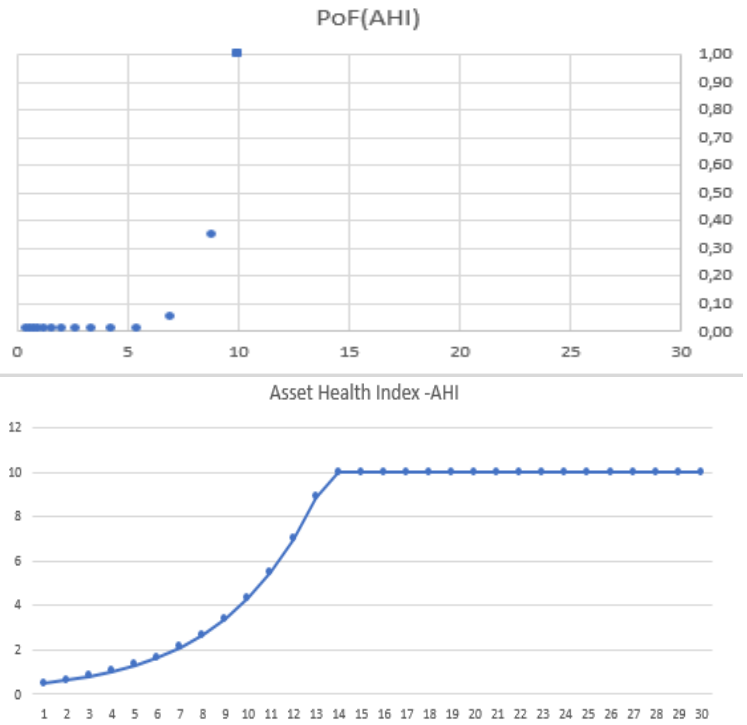


Figure 45: AHI monitoring and Pof representation

The results of the ANN, have already been exposed before in the thesis. It has been made the validation of the training with different number of neurons; and with Data that is not related to the work but has been useful to validate the work of the ANN. So the training and results of the ANN are validated with a very small error, but the absence of the real data of the operational factors, made it impossible to implement the ANN on the Matlab Simulink DT model. So the ANN model is proposed, with it´s structure, inputs, and outputs; and it will only need aviable and real training data to be implemented in the DT.

CHAPTER 7 CONCLUSIONS

The focus of this thesis lies on the DT concept and the application of the condition based maintenance and neural networks to the digital field. It is a highly relevant concept in the recent industrial and research initiatives level. The research started from the current classic simulation practise in the manufacturing world to what enterprises may benefit from the adoption of DT in their business, shaping the new role of DT in the Industry 4.0 paradigm.

Although it draws big attentions in the research field, many studies and works did not necessarily share and reflect the same definition of DT. In this end, this work first of all proposes a literature analysis about the concept of DT by offering a systematic review of what has been written on this topic in the manufacturing domain. This step is necessary and important because it sets the starting point and the basis of the following research works in this elaboration. After the work three clear interpretations of the DT function are concluded:

- DT as a virtual monitoring of the physical variables.
- DT as a decision taking tool, based on the condition of the machine given by the variables of the digital model.
- As a virtual commissioning tool, that can allow to make a digital prototype that does the same functions as a physical

After having these uses of the DT defined, it could be easily seen that this work focuses on the monitoring and decision taking aspects. The monitoring objective is reached by the connection of the computer tools with the PLC, and the decision taking aspect has been developed and proposed but not validated because of need of a tool to send the control actions from the computer back to the PLC, and the complexity of this need.

So it can be concluded that, if the PLC and the MES provide the signals and the production plan with the operation mechanical factors; the states of the machine, the condition of the asset health, and the probability of failure can be provided by the DT in real time and it allows a huge world of decision taking possibilities to control the performance of the machine studied. It is only needed to develop a way or tool to send the control decisions back to the PLC in order to make them effective on the field.

Apart from that, this work also gives an interesting conclusion about the new tools that are born in the recent days, and are focused to vigilance and machine learning.

The methods related to the AHI and condition based maintenance are applied to reach a model that controls the health of the asset and the probability of afailure, but with the application of the ANN there is a different point of view.

The ANN applied to the DT, can allow to have a tool that will detect any variation on the asset performance, but it is not based on probability and can appreciate any anomaly considering a much deeper level. So to conclude, in a further work, it could be reached to make a DT model of a system that analyses and trains itself, that will allow to have a total control of the asset performance which could be more energetically efficient and also a lot cheaper to maintain from the bussines point of view.

REFERENCES

[1] W. MacDougall. *Industrie 4.0: Smart manufacturing for the future*. Germany Trade & Invest, 2014.

[2] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, and M. Hoffmann. Industry 4.0. *Business & Information Systems Engineering*, 6(4):239–242, 2014.

[3] H. Kagermann, W.-D. Lukas, and W. Wahlster. Industrie 4.0: Mit dem internet der dinge auf dem weg zur 4. industriellen revolution. *VDI nachrichten*, 13:11, 2011.

[4] A. A. F. Saldivar, Y. Li, W.-n. Chen, Z.-h. Zhan, J. Zhang, and L. Y. Chen. Industry 4.0 with cyber-physical integration: A design and manufacture perspective. In *Automation and computing (icac), 2015 21st international conference on*, pages 1–6. IEEE, 2015.

[5] M. Rüßmann, M. Lorenz, P. Gerbert, M. Waldner, J. Justus, P. Engel, and M. Harnisch. Industry 4.0: The future of productivity and growth in manufacturing industries. *Boston Consulting Group*, 9, 2015.

[6] R. H. Weber and R. Weber. Internet of things, volume 12. Springer, 2010.

[7] J. Banks and J. S. Carson. Ii. 1984. Discrete-event system simulation.

[8] S. H. Khajavi and J. Holmström. Manufacturing digitalization and its effects on production planning and control practices. In *IFIP International Conference on Advances in Production Management Systems*, pages 179–185. Springer, 2015.

[9] R. B. Detty and J. C. Yingling. Quantifying benefits of conversion to lean manufacturing with discrete event simulation: a case study. *International Journal of Production Research*, 38(2):429–445, 2000.

[10] J. S. Smith. Survey on the use of simulation for manufacturing system design and operation. *Journal of manufacturing systems*, 22(2):157–171, 2003.

[11] A. Jamalnia and A. Feili. A simulation testing and analysis of aggregate production planning strategies. *Production Planning & Control*, 24(6):423–448, 2013.

[12] T. Yang. An evolutionary simulation–optimization approach in solving parallel-machine scheduling problems–a case study. *Computers & Industrial Engineering*, 56(3):1126–1136, 2009.

[13] G. Metan, I. Sabuncuoglu, and H. Pierreval. Real time selection of scheduling rules and knowledge extraction via dynamically controlled

data mining. *International Journal of Production Research*, 48(23):6909–6938, 2010.

[14] U. Ghani, R. Monfared, and R. Harrison. Real time energy consumption analysis for manufacturing systems using integrative virtual and discrete event simulation. 2012.

[15] C. Herrmann, S. Thiede, S. Kara, and J. Hesselbach. Energy oriented simulation of manufacturing systems–concept and application. *CIRP Annals-Manufacturing Technology*, 60(1):45–48, 2011.

[16] S. Thiede, Y. Seow, J. Andersson, and B. Johansson. Environmental aspects in manufacturing system modelling and simulation—state of the art and research perspectives. *CIRP Journal of manufacturing science and technology*, 6(1): 78–87, 2013.

[17] N. Weinert, S. Chiotellis, and G. Seliger. Methodology for planning and operating energy-efficient production systems. CIRP Annals-Manufacturing Technology, 60(1):41–44, 2011.

[18] M. Taisch, B. Stahl, F. Vaccari, and A. Cataldo. A production-state based approach for energy flow simulation in manufacturing systems. In *IFIP International Conference on Advances in Production Management Systems*, pages 227–234. Springer, 2013.

[19] M. Grieves. Digital twin: Manufacturing excellence through virtual factory replication. *White paper*, 2014.

[20] A. Canedo. Industrial iot lifecycle via digital twins. In *Proceedings of the Eleventh IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*, page 29. ACM, 2016.

[21] T. Gabor, L. Belzner, M. Kiermeier, M. T. Beck, and A. Neitz. A simulation-based architecture for smart cyber-physical systems. In *Autonomic Computing (ICAC), 2016 IEEE International Conference on*, pages 374–379. IEEE, 2016.

[22] M. Schluse and J. Rossmann. From simulation to experimentable digital twins: Simulation-based development and operation of complex technical systems. In *Systems Engineering (ISSE), 2016 IEEE International Symposium on*, pages 1–6. IEEE, 2016.

[23] R. Rosen, G. Von Wichert, G. Lo, and K. D. Bettenhausen. About the importance of autonomy and digital twins for the future of manufacturing. *IFAC-PapersOnLine*, 48(3):567–572, 2015.

[24] J. Lee, E. Lapira, B. Bagheri, and H.-a. Kao. Recent advances and trends in predictive manufacturing systems in big data environment. *Manufacturing Letters*, 1(1):38–41, 2013.

[25] S. Boschert and R. Rosen. Digital twin—the simulation aspect. In

Mechatronic Futures, pages 59–74. Springer, 2016.

[26] R. A. Rayan and A. Kaviya. Digital twins: A lead to industrial revolution.

[27] P. Hirmer, U. Breitenbu¨cher, A. C. F. da Silva, K. K´epes, B. Mitschang, and M. Wieland. Automating the provisioning and configuration of devices in the internet of things. Complex Systems Informatics and Modeling Quarterly, (9): 28–43, 2016.

[28] G. N. Schroeder, C. Steinmetz, C. E. Pereira, and D. B. Espindola. Digital twin data modeling with automationml and a communication methodology for data exchange. IFAC-PapersOnLine, 49(30):12–17, 2016.

[29] S. Weyer, T. Meyer, M. Ohmer, D. Gorecky, and D. Zühlke. Future modeling and simulation of cps-based factories: an example from the automotive industry. *IFAC-PapersOnLine*, 49(31):97–102, 2016.

[30] T. Wuest, K. Hribernik, and K.-D. Thoben. Capturing, managing and sharing product information along the lifecycle for design improvement. In *Proceedings of the 10th International Workshop on Integrated Design Engineering*, 2014.

[31] H. Kagermann, W.-D. Lukas, and W. Wahlster. Industrie 4.0: Mit dem internet der dinge auf dem weg zur 4. industriellen revolution. *VDI nachrichten*, 13:11, 2011.

[32] G. Schroeder, C. Steinmetz, C. E. Pereira, I. Muller, N. Garcia, D. Espindola, and R. Rodrigues. Visualising the digital twin using web services and augmented reality. In *Industrial Informatics (INDIN), 2016 IEEE 14th International Conference on*, pages 522–527. IEEE, 2016.

[33] J. Lee, B. Bagheri, and H.-A. Kao. A cyber-physical systems architecture for industry 4.0-based manufacturing systems. *Manufacturing Letters*, 3:18–23, 2015.

[34] R.J Heywood,T. McGrail “Generating Asset Health Indices Which Are Useful and Auditable” 2015: 1-6.

[35] Naderian A, Cress S, Piercy R, Wang F, Service J. An Approach to Determine the Health Index of Power Transformers. Conference Record of the IEEE International Symposium on Electrical Insulation. 2008; July: 192–196.https://doi.org/10.1109/ELINSL.2008.4570308

[36] Okamotoa, M., Nonakaa, T., Ochiaia S. and Tominagaa, D. 1998. Nonlinear numerical optimization with use of a hybrid Genetic Algorithm incorporating the Modified Powell method. Applied Mathematics and Computation. 91(1). 63-72.

[37] Powell, M.J.D.1964. An efficient method for finding the minimum of a function of several variables without calculating derivatives. Computer Journal. 7(2). 155-62.

[38] Powell, M.J.D.1968. On the calculation of orthogonal vectors. Computer Journal. 11(2). 302-304.

[39] "Advanced Prognostics for Smart Systems" . IMS Center. Retrieved 23 January2014.

[40] S. Mousavi, S. Thiede, W. Li, S. Kara, and C. Herrmann. An integrated approach for improving energy efficiency of manufacturing process chains. *International Journal of Sustainable Engineering*, 9(1):11–24, 2016.

[41] N. Weinert, S. Chiotellis, and G. Seliger. Methodology for planning and operating energy-efficient production systems. CIRP Annals-Manufacturing Technology, 60(1):41–44, 2011.

[42] Scatiggio, F., Pompili, M. Health index: the TERNA's practical approach for transformers fleet management. IEEE Electrical Insulation Conf. (EIC). 2013: 178–182

[43] Scatiggio F, Rebolini M, Pompili M. Health Index: The Last Frontier of TSO's Asset Management. Terna Rete. 2016: 1–9.

[44] Pompili M, Scatiggio F. Classification in Iso-Attention Classes of Hv Transformer Fleets." IEEE Transactions on Dielectrics and Electrical Insulation. 2015; 22(5):2676–83.

APPENDIX (MATBLAB/OPC UA/VENSIM CODES)

Matlab Simulink variables in real time

On Matlab Simulink progamation of a DT model functions that work in real time, all the variables whose values change meanwhile the simulation is running should be declared in the model code as “persistent”

```
persistent IdleTime
if isempty(IdleTime)
IdleTime=0;
end
```

This way, the variable persists during the simulation time (Which is real time) and actualizes it´s value.

Level 2 S-Function of Drilling Module

```
function Drilling(block)
   setup(block);

function setup(block)
  % Some example set up
  % Register the number of ports.
  block.NumInputPorts  = 0;
  block.NumOutputPorts = 3;

  % Set up the port properties to defaults
  block.SetPreCompPortInfoToDefaults;

  %% Set block sample time to [0.4 0]
 block.SampleTimes = [0.4 0];

  % Set output port dimensions (depends on number of
channels)
  block.OutputPort(1).DatatypeID  = 8;
  block.OutputPort(1).Complexity  = 'Real';
  block.OutputPort(1).Dimensions = 1;

  block.OutputPort(2).DatatypeID  = 8;
  block.OutputPort(2).Complexity  = 'Real';
  block.OutputPort(2).Dimensions = 1;

  block.OutputPort(3).DatatypeID  = 4;
  block.OutputPort(3).Complexity  = 'Real';
  block.OutputPort(3).Dimensions = 1;
  % Register functions
  block.RegBlockMethod('Start', @Start);
  block.RegBlockMethod('PostPropagationSetup',
@DoPostPropSetup);
  block.RegBlockMethod('Outputs', @Outputs);
  block.RegBlockMethod('Terminate', @Terminate);
```

```
function DoPostPropSetup(block)


function Start(block)
%connection
serverList = opcuaserverinfo('10.48.134.30:4840')
uaClient = opcua(serverList)
connect(uaClient)
%find the pointer to the signal needed
f=getNamespace(uaClient)
b=findNodeByName(f, 'plcDrillingCPS','-once')
d=findNodeByName(b, 'Outputs','-once')
m=findNodeByName(b, 'Inputs','-once')
k=findNodeByName(b, 'DataBlocksGlobal','-once')
j=findNodeByName(k, 'dbAppIF','-once')
p=findNodeByName(j, 'Out','-once')
g=findNodeByName(d, 'xQA1_A1','-once') %conveyor
h=findNodeByName(m, 'xBG1','-once') %start working
sensor
l=findNodeByName(p,'iRetCode','-once')
myData = [g, h, l];
set_param(block.BlockHandle,'UserData', myData)


function Outputs(block)
    data = get_param(block.BlockHandle,'UserData');
    stored_g = data(1);
    stored_h = data(2);
    stored_l = data(3);
  block.OutputPort(1).Data = readValue(stored_g);
  block.OutputPort(2).Data = readValue(stored_h);
  block.OutputPort(3).Data = readValue(stored_l);

function Terminate(block)
% Get the userdata
handle = get_param(block.BlockHandle,'UserData');
% Do some termination routines with the handle
% Clear the UserData
set_param(block.BlockHandle,'UserData',[])
```

Level 2 S-Function "DataTypeID"

Information on the DatatypeIDs associated with Level 2 MATLAB file S-functions is missing from the Simulink documentation. DatatypeID of an input port must be one of the following:

-1 for 'inherited'
0 for 'double'
1 for 'single'
2 for 'int8'
3 for 'uint8'
4 for 'int16'
5 for 'uint16'
6 for 'int32'
7 for 'uint32'
8 for 'boolean'

Time Accumulator Matlab Function

```
function  Accumulator(y,Time)
%Variables that acumulate the times in each state

persistent IdleTime
if isempty(IdleTime)
IdleTime=0;
end
persistent IdleTimeAccum
if isempty(IdleTimeAccum)
IdleTimeAccum=0;
end
persistent WorkingTime
if isempty(WorkingTime)
WorkingTime=0;
end
persistent WorkingTimeAccum
if isempty(WorkingTimeAccum)
WorkingTimeAccum=0;
end
persistent EnergySavingTime
if isempty(EnergySavingTime)
EnergySavingTime=0;
end
persistent EnergySavingTimeAccum
if isempty(EnergySavingTimeAccum)
EnergySavingTimeAccum=0;
end
persistent FailureTime
if isempty(FailureTime)
FailureTime=0;
end
persistent FailureTimeAccum
if isempty(FailureTimeAccum)
FailureTimeAccum=0;
end

%(Loop condition that keeps it working during the simulation time)

switch y
    case 1
        IdleTime=IdleTime+1;
        IdleTimeAccum=IdleTime*0.4;
    case 2
        WorkingTime=WorkingTime+1;
        WorkingTimeAccum=WorkingTime*0.4;
    case 3
        EnergySavingTime=EnergySavingTime+1;
        EnergySavingTimeAccum=0.4*EnergySavingTime;
    otherwise
        FailureTime=FailureTime+1;
        FailureTimeAccum=0.4*FailureTime;
end
persistent fileID
if isempty(fileID)
fileID = fopen('Accumulation.txt','w');
fprintf(fileID,'%16s %16s %16s %16s\r
','IdleTime','WorkingTime','EnergySavingTime','FailureTime');
```

```
end
fprintf(fileID,'\r\n%12.8f %12.8f %12.8f
%12.8f\n\r',IdleTimeAccum,WorkingTimeAccum,EnergySavingTimeAccum,Failu
reTimeAccum);
end
```

Drilling Machine Condition Model Matlab Implementation

```
function  MachineCondition(Time,state,hardness,drill)

persistent IdleTime
if isempty(IdleTime)
IdleTime=0;
end
persistent IdleTimeAccum
if isempty(IdleTimeAccum)
IdleTimeAccum=0;
end
persistent WorkingTime
if isempty(WorkingTime)
WorkingTime=0;
end
persistent WorkingTimeAccum
if isempty(WorkingTimeAccum)
WorkingTimeAccum=0;
end
persistent EnergySavingTime
if isempty(EnergySavingTime)
EnergySavingTime=0;
end
persistent EnergySavingTimeAccum
if isempty(EnergySavingTimeAccum)
EnergySavingTimeAccum=0;
end
persistent FailureTime
if isempty(FailureTime)
FailureTime=0;
end
persistent FailureTimeAccum
if isempty(FailureTimeAccum)
FailureTimeAccum=0;
end
persistent AOT
if isempty(AOT)
AOT=0;
end
persistent AHI
if isempty(AHI)
AHI=0;
end
persistent PoF
if isempty(PoF)
PoF=0;
end
% persistent Preventive
% if isempty(Preventive)
% Preventive=0;
% end
randomnumber=rand;
```

```
persistent fileID
if isempty(fileID)
fileID = fopen('MachineCondition.txt','w');
fprintf(fileID,'%6s %12s\n\r','Time','AssetHealth');
end

switch state
    case 1
        IdleTime=IdleTime+1;
        IdleTimeAccum=IdleTime*0.4;
        DH=0;
        AV=(IdleTimeAccum + WorkingTimeAccum +
EnergySavingTimeAccum)/(IdleTimeAccum + WorkingTimeAccum +
EnergySavingTimeAccum + FailureTimeAccum);
        UT= WorkingTimeAccum / (IdleTimeAccum + WorkingTimeAccum +
EnergySavingTimeAccum);
        AR=(log(5.5/0.5))/10;
        AOT= AOT+DH;
        AHI=0.5*exp(AR*AOT);
         if AHI <=5.5
            PoF=0.01;
         else if AHI<10
                PoF=0.0000359381*exp(1.02337*AHI);
             else
                PoF=1;
            end
         end
       fprintf(fileID,'\r\n%6.2f %12.8f\n\r',Time,AHI);
        if randomnumber>PoF
            Fail=1;
%             state=4;
        else
            Fail=0;
        end
%         FailurePolicy=;
%
%         if PoF>FailurePolicy
%
%             Preventive=1;
%         else
%             Preventive=0;
%         end
%
    case 2
        WorkingTime=WorkingTime+1;
        WorkingTimeAccum=WorkingTime*0.4;
        DH=hardness*drill;
        AV=(IdleTimeAccum + WorkingTimeAccum +
EnergySavingTimeAccum)/(IdleTimeAccum + WorkingTimeAccum +
EnergySavingTimeAccum + FailureTimeAccum);
        UT= WorkingTimeAccum / (IdleTimeAccum + WorkingTimeAccum +
EnergySavingTimeAccum);
        AOT= AOT+DH;
        AR=(log(5.5/0.5))/10;
        AHI=0.5*exp(AR*AOT);
        if AHI <=5.5

            PoF=0.01;
        else if AHI<10
                PoF=0.0000359381*exp(1.02337*AHI);
```

```
            else
                PoF=1;
            end
        end
        fprintf(fileID,'\r\n%6.2f %12.8f\n\r',Time,AHI);
        if randomnumber>PoF
            Fail=1;
%             state=4;
        else
            Fail=0;
        end
%         FailurePolicy=;
%
%         if PoF>FailurePolicy
%
%             Preventive=1;
%         else
%             Preventive=0;
%         end

    case 3

        EnergySavingTime=EnergySavingTime+1;
        EnergySavingTimeAccum=0.4*EnergySavingTime;
        DH=0;
        AV=(IdleTimeAccum + WorkingTimeAccum +
EnergySavingTimeAccum)/(IdleTimeAccum + WorkingTimeAccum +
EnergySavingTimeAccum + FailureTimeAccum);
        UT= WorkingTimeAccum / (IdleTimeAccum + WorkingTimeAccum +
EnergySavingTimeAccum);
        AR=(log(5.5/0.5))/10;
        AOT= AOT+DH;
        AHI=0.5*exp(AR*AOT);
         if AHI <=5.5

            PoF=0.01;

        else if AHI<10

                PoF=0.0000359381*exp(1.02337*AHI);
            else
                PoF=1;
            end
         end

        fprintf(fileID,'\r\n%6.2f %12.8f\n\r',Time,AHI);

        if randomnumber>PoF

            Fail=1;
%             state=4;
        else
            Fail=0;
        end

%         FailurePolicy=0.7;
%
%         if PoF>FailurePolicy
%
%             Preventive=1;
```

```
%          else
%              Preventive=0;
%          end
%
    otherwise

        FailureTime=FailureTime+1;

        FailureTimeAccum=0.4*FailureTime;

        DH=0;
        AV=(IdleTimeAccum + WorkingTimeAccum +
EnergySavingTimeAccum)/(IdleTimeAccum + WorkingTimeAccum +
EnergySavingTimeAccum + FailureTimeAccum);
        UT= WorkingTimeAccum / (IdleTimeAccum + WorkingTimeAccum +
EnergySavingTimeAccum);
        AOT=0;
        AR=(log(5.5/0.5))/10;
        AHI=0.5*exp(AR*AOT);
        fprintf(fileID,'\r\n%6.2f %12.8f\n\r',Time,AHI);

%          FailurePolicy=0.7;
%
%          if PoF>FailurePolicy
%
%              Preventive=1;
%          else
%              Preventive=0;
%          end

end
end
```

Matlab OPC UA Toolbox Functions

- ***opcuaserverinfo ('HostName'):*** queries the host named HostName for its installed OPC UA servers. HostName can be a host name, or IP address, specified as a character vector or string. It returned an OPC UA ServerInfo object, or an array of these objects, containing the read-only properties Description, Hostname, and Port.

- ***opcua(ServerInfoObj/ServerUrl/Hostname,Portnum):*** constructs an OPC UA client object with the server referenced by ServerInfoObj/ServerUrl/Portnum,Hostname.

- ***connect(UaClient):*** connects the OPC UA client UaClient to its referenced server using anonymous user authentication.

- ***disconnect(UaClient):*** disconnects the OPC UA client UaClient from its server.
- ***getNamespace(UaClient):*** retrieves one layer of the namespace of the server associated with client object UaClient. The namespace is stored in the Namespace property of uaClient as a hierarchical tree of nodes.

- ***browseNamespace (opcua):*** opens the Browse Name Space dialog box for OPC UA client object UaClient. Using this browser, you can construct a list of nodes, and return an array of those nodes in NodeList.
  ftndNodeByName(NodeList,NodeName)

  ftndNodeByName(NodeList,NodeName,'-once') : searches the descendants of NodeList for all nodes whose Name property matches NodeName. The search among all nodes, including NodeList, is not case sensitive. findNodeByName(NodeList,NodeName,'-once') stops searching when one node has been found.

- ***readValue(UaClient,NodeList):*** reads the value, quality, and timestamp from the nodes identified by NodeList, on the server associated with the connected client UaClient.NodeList can be a single OPC UA node object or an array of nodes.

- ***writeValue(UaClient,NodeList,Values*)**: writes content of Values, to the nodes identified by NodeList. You can browse for node objects using browseNamespace. You can also create nodes using opcuanode. If NodeList is a single node, then Values is the value written to the node. If NodeList is an array of nodes, Values must be a cell array the same size as NodeList, and each element of the cell array is written to the corresponding element of NodeList.

Vensim Simulation Model Code

```
********************************************************
    .Subscript
********************************************************~|
Input Signal:       Input a, Input b, Input c    ~~|
Neurons in the Hyden Layer:         First, Second, Third, Forth, Fifth, Sixth        ~~|
********************************************************
    .Auxiliary
********************************************************~|
Delta H1[Neurons in the Hyden Layer]= Delta O*W2[Neurons in the Hyden Layer]*OutH[Neurons in the
    Hyden Layer]*(1-OutH[Neurons in the Hyden Layer])       ~~|
Delta O=-(Target-OutO)*OutO*(1-OutO)      ~~|
Input[Input Signal]=Inputs Data[Input Signal](Time)    ~~|
NetH[Neurons in the Hyden Layer]= SUM(Input[Input Signal!]*W1[Input Signal!,Neurons in the Hyden
    Layer])+Bias 1          ~~|
NetO=SUM(OutH[Neurons in the Hyden Layer!]*W2[Neurons in the Hyden Layer!])+Bias 2          ~~|
OutH[Neurons in the Hyden Layer]=           1/(1+EXP(-NetH[Neurons in the Hyden Layer])) ~~|
OutO=   1/(1+EXP(-NetO)) ~~|
Target=Target Data(Time)  ~~|
********************************************************
    .Constant
********************************************************~|
Eta= 0.934          ~~|
Bias 1=   0         ~~|
Bias 2=   0.0659 ~~|
********************************************************
    .Level
********************************************************~|
W2[Neurons in the Hyden Layer]= INTEG (-Adj2[Neurons in the Hyden Layer], 1)          ~~|
W1[Input Signal,Neurons in the Hyden Layer]=
    INTEG (-Adj1[Input Signal,Neurons in the Hyden Layer], 0.5)          ~~|
Acc Square Error= INTEG (Current SQ Error,0)          ~~|
********************************************************
    .Flow
********************************************************~|
Current SQ Error=(1/2)*((Target-OutO)^2)    ~~|
Adj1[Input Signal,Neurons in the Hyden Layer]=
    Delta H1[Neurons in the Hyden Layer]*Input[Input Signal]*Eta      ~~|
```

Adj2[Neurons in the Hyden Layer]=

Delta O*OutH[Neurons in the Hyden Layer]*Eta ~~|

****

.Data

**********~|

Inputs Data[Input a]([(0,0)(200,1)], (0,0.5),(49.2872,0.757143), (74.5417,0.390476), (100,0.5), (129.939, 0.642857), (150.305, 0.247619), (200, 0.5)) ~~|

Inputs Data[Input b]([(0,0)-(200,1)], (0,0.5), (21.1813,0.395238), (70.4684,0.352381), (100,0.5), (122.2, 0.614286), (173.116, 0.633333), (200,0.5)) ~~|

Inputs Data[Input c]([(0,0)-(200,1)], (0,0), (17.1079,0.347619), (74.5417,0.347619), (100,0.5), (124.644, 0.357143), (167.413, 0.319048), (200, 0.5)) ~~|

Target Data([(0,0)-(200,1)], (0,0.5), (12.0163, 0.357143), (25.2546, 0.252381), (37.4745, 0.37619), (49.8982 ,0.371429), (60.2851, 0.369668), (74.9491, 0.490476), (88.7984, 0.350711), (100, 0.5), (115.275, 0.628571), (135.642, 0.671429), (152.749, 0.528571), (175.153, 0.566667), (200,0.8)) ~~|

****

.Control

**********~

Simulation Control Parameters |

FINAL TIME = 200 ~Unit time ~The final time for the simulation. |

INITIAL TIME = 0 ~Unit Time~The initial time for the simulation. |

SAVEPER = TIME STEP ~Unit Time ~The frequency with which output is stored|

TIME STEP = 1 ~Unit Time ~The time step for the simulation.|