Integration of ROS and Tecnomatix for the Development of Digital Twins Based Decision-Making Systems for Smart Factories

Carolina Saavedra Sueldo, Sebastián A. Villar, Mariano De Paula and Gerardo G. Acosta

***Abstract*—Digital twins employs simulation in conjunction with virtual environments and a variety of data coming from different plant equipment and physical systems to continuously update the digital models of the world in a feedback loop scheme to facilitate the decision-making processes. The heterogeneity of existing hardware and software requires the development of software architectures able to deal with the information exchange due to the integration and interaction of several system components and autonomous decision-making systems. In this work we propose the design and construction of a software architecture that integrates a manufacturing process simulator with the well-known robot operating system (ROS-Robot Operating System) to easily interchange information with an autonomous decision-making system. The proposal is tested with the simulator Tecnomatix® and the free distribution ROS Melodic. We present an instance of software architecture for a typical complex case study of manufacturing plants and demonstrate its easy integration with an autonomous decision-making system based on the reinforcement- learning paradigm.**

***Index Terms*—Digital Twin, Autonomous Decision System, Integration, Industry 4.0, ROS, Tecnomatix.**

I. Introducción

Los modos de producción actuales se están moviendo hacia una filosofía de producción personalizada en la que se acentúa cada vez más la utilización de recursos y/o servicios rentados dentro de los sistemas productivos para tratar de conseguir una alta flexibilidad y capacidad de reconfiguración [1]. Esto hace que los sistemas sean cada vez más complejos y con una mayor tendencia a incorporar otros sub-sistemas automatizados y/o robóticos para la realización de una multiplicidad de tareas a lo largo de todo el proceso productivo [2]. Sin embargo, en esta nueva configuración de sistemas de producción, existen inevitablemente interferencias dinámicas asociadas con diversas fuentes (p.e., la aleatoriedad de los pedidos, la baja repetitividad del proceso de producción, el uso de servicios y recursos externos, etc) que hacen que la producción sincronizada pueda verse perturbada en detrimento de la efectividad de respuesta del sistema productivo [3].

Carolina Saavedra Sueldo, INTELYMEC, Centro de Investigaciones en Física e Ingeniería del Centro –UNICEN – CICpBA – CONICET, B7400JWI, Olavarría, Argentina (e-mail: carolina.saavedra@fio.unicen.edu.ar).

Sebastián A. Villar, IEEE Member, INTELYMEC, Centro de Investigaciones en Física e Ingeniería del Centro –UNICEN – CICpBA – CONICET, B7400JWI, Olavarría, Argentina (e-mail: svillar@fio.unicen.edu.ar).

Mariano De Paula, INTELYMEC, Centro de Investigaciones en Física e Ingeniería del Centro –UNICEN – CICpBA – CONICET, B7400JWI, Olavarría, Argentina (e-mail: mariano.depaula@fio.unicen.edu.ar).

Gerardo G. Acosta, IEEE Member, INTELYMEC, Centro de Investigaciones en Física e Ingeniería del Centro –UNICEN – CICpBA – CONICET, B7400JWI, Olavarría, Argentina (e-mail: ggacosta@fio.unicen.edu.ar).

En las configuraciones actuales de los sistemas productivos modernos intervienen una variedad de equipos y recursos de software y hardware y, también, recursos humanos que deben actuar coordinadamente y de manera autónoma para hacer frente a la variación y alta frecuencia de las tareas de producción causadas por los requisitos personalizados de los clientes [4]. En este contexto, el paradigma de simulación clásico (dinámica y de eventos discretos) ha dejado de ser *per se* el componente central para la formulación de sistemas autónomos de toma de decisiones debido, principalmente, a su característica estática. Por ello, y dada la fuerte irrupción de internet industrial de las cosas [5], [6] junto a los sistemas ciberfísicos [7], su lugar ha sido ocupado por el concepto de gemelo digital (GD) [8].

Si bien, desde hace tiempo, existe una tecnología estable en lo que se refiere a cada uno de los componentes individuales que conforman un sistema de manufactura flexible, la integración entre las diferentes tecnologías para el desarrollo de los GDs es una cuestión que no ha sido abordada profusamente [9] y que es de vital relevancia para el desarrollo de propuestas de sistemas autónomos de manufactura flexible. En este trabajo, se presenta una arquitectura software para el desarrollo de GDs y su integración con un sistema autónomo de toma de decisiones. Dicha arquitectura de software se centra en la integración del bien conocido sistema operativo de robots (ROS) y un simulador de procesos de manufactura flexible, en este caso el *Tecnomatix Plant Simulation* – Siemens®(TPS).

II. Trabajos Relacionados

Para analizar el comportamiento y el desempeño de los sistemas de manufactura, la simulación se ha adoptado como la herramienta de referencia para representar sistemas complejos a través de modelos simples y fáciles de comprender [10]. Una variedad de modelos de simulación han sido construidos como base para la toma de decisiones en la industria de manufactura [11]. Esto ha convertido a la simulación en una de las herramientas utilizadas para analizar el impacto de las decisiones tomadas sin afectar la operación del sistema real [12].

En muchos casos se ha utilizado la simulación en la formulación de problemas de tomas de decisiones en sistemas industriales como problemas de optimización [13] incluyendo también aquellos multi-objetivo [14]. Por ejemplo, para el caso de un problema de asignación de recursos en un sistema productivo, que considera la incertidumbre y la flexibilidad

de la secuencia de trabajos, se planteó un problema de optimización multi-objetivo y se desarrolló y usó un modelo de simulación de eventos discretos del proceso productivo para encontrar una solución a dicho problema [15]. Por otro lado, la simulación computacional también ha sido usada para la identificación de parámetros óptimos como el tamaño del lote, el stock, la secuencia de procesamiento de órdenes de trabajo o la frecuencia de entrega [16]. Además, en los modelos de simulación se pueden incluir cambios externos mientras el sistema actualiza los parámetros de forma autónoma [17], [18].

El desarrollo confiable y seguro de internet industrial de las cosas [6] ha potenciado la interoperabilidad de los sistemas de manufactura [19] al mismo tiempo que ha facilitado la interacción entre los diferentes componentes y sub-sistemas que componen cada sistema productivo. En este sentido, las posibilidades tecnológicas actuales han impulsado el concepto de GD [20]. Sin embargo, en el desarrollo de los GDs los métodos y técnicas de simulación de sistemas tienen un papel preponderante para la abstracción y representación de los sistemas productivos [21]. En este sentido, los GDs emplean la simulación en un esquema de intercambio de información con una variedad de datos provenientes de los diferentes equipos y sistemas físicos, para mantener actualizados continuamente los modelos digitales con el fin de reflejar cualquier cambio que ocurra en los equivalentes físicos en el tiempo [22]. De esta manera, se puede lograr un esquema de retroalimentación en un entorno virtual adaptativo que facilita la toma de decisiones apoyándose en estos GDs. Bajo circunstancias y condiciones de incertidumbre los sistemas de producción pueden beneficiarse ampliamente por el enfoque de control, monitoreo y supervisión basado en el uso de GDs para obtener la información completa de los elementos y componentes necesarios para la toma de decisiones [23].

Sin embargo, para todas las propuestas que involucren el uso de GDs es imperioso el desarrollo de arquitecturas de integración que permitan la interacción entre los sub-sistemas físicos involucrados y los cibercomponentes para dar lugar a la creación de tales GDs. En este sentido, se han realizado pocas propuestas, como por ejemplo, en el reciente trabajo de [24] donde presenta el uso de un método de sincronización basado en simulación para la validación virtual de sistemas de producción en la fase de planificación. Para ello, se utiliza una aplicación de software en el que el gemelo digital del proceso de fabricación virtual se divide en un gemelo digital del proceso relacionado con el producto y en otro relacionado con los recursos. La metodología se verifica utilizando un simulador de una estación de ensamblaje automatizado del chasis del automóvil para el alivio físico de los trabajadores en el área de ensamblaje final. En el trabajo de [25] se propone un mecanismo de colaboración inteligente para determinar la configuración óptima de los recursos en un sistema de ensamblaje y asumiendo simplemente que la intercomunicación entre los diferentes recursos se establece mediante Internet Industrial de los datos recogidos por lectores RFID y transmitidos por chips NFC (*Near Field Communication*). Similarmente, en el trabajo [26] proponen un método de reconfiguración automático del sistema de manufactura basado en el uso de un GD y para el intercambio de datos desarrollan una máquina herramienta de arquitectura abierta que se define y desarrolla como una nueva clase de máquina herramienta que comprende una plataforma estándar fija y varios módulos individualizados que se pueden agregar e intercambiar. Sin embargo, esta arquitectura presenta limitaciones cuando intervienen sistemas robóticos en el proceso productivo. Recientemente los autores de [27] presentaron una arquitectura modular para implementar un GD para un caso de una línea de producción comandada mediante un PLC para la industria de procesos mientras que en [9] se propone una arquitectura colaborativa, para una plataforma industrial de Internet, denominada sistema operativo industrial (SO-industrial) que aloja al controlador industrial, el hilo digital y los micro-servicios para lograr un entorno de sistema de información empresarial cooperativo para los sistemas de fabricación.

Como se puede apreciar en los últimos trabajos citados, la integración digital de las partes físicas y de software para el desarrollo de GDs de los sistemas productivos es una cuestión que está recientemente recibiendo una especial atención. Sin embargo, para el caso de sistemas productivos en los que intervienen diversos sistemas robóticos, y de diferente naturaleza, las propuestas de arquitecturas de integración son prácticamente nulas, cuestión que motiva el objetivo principal para el desarrollo de este trabajo.

III. Metodología

En esta sección introducimos muy brevemente los principales componentes metodológicos hacia la formulación de un GD para un sistema de manufactura flexible.

A. Simulación de Eventos Discretos

La simulación de eventos discretos es una herramienta de análisis ampliamente difundida en el ambiente empresarial para apoyar la toma de decisiones [11], [12], [28]. Brevemente, esta herramienta permite simular situaciones complejas de la realidad bajo análisis permitiendo extraer información y datos relevantes para la toma de decisiones, en busca de reducir la incertidumbre y riesgo de las decisiones realmente ejecutadas.

Los modelos de simulación representan procesos y situaciones complejas de la realidad en función de los eventos que puedan ocurrir por lo que pueden representar tanto situaciones típicas de un proceso productivo (rotura de inventarios, fallas de equipos, aparición de imponderables, entre otros), como así también estimar un determinado nivel de utilización de recursos relacionados con dichas situaciones. Asimismo, contempla las probabilidades de que ocurran cada uno de los eventos según se asocian con las características de cada uno de los elementos que intervienen. En su conjunto, todos estos elementos simulan la evolución natural del proceso analizado. De esta forma los modelos permiten conceptualizar sobre un proceso productivo y su gestión en términos de los eventos que puedan suceder y cuyo impacto afecta tanto a los clientes como a otros componentes del sistema (p. ej., el uso de recursos).

B. Génesis de los Gemelos Digitales

Gracias a las tecnologías actuales es posible la integración de todas las entidades que intervienen en un sistema productivo [29]. Esta integración ciberfísica está siendo cada

vez más adoptada involucrando conexiones intensivas y flujos de intercambio de información entre las distintas entidades físicas y los sistemas computacionales. De esta forma, con esta integración y la interacción en tiempo real es posible monitorear y controlar entidades físicas de manera confiable, segura, colaborativa, robusta y eficiente [21]. Entonces, los sistemas de fabricación integrados podrían pensarse compuestos por dos partes principales: una parte física y una parte digital. Como se muestra en la Fig. 1, la parte física comprende todas las entidades y recursos empleados para la fabricación mientras que la parte digital contiene aplicaciones y servicios, incorporando subsistemas con capacidades para el procesamiento y gestión de datos, simulación y análisis junto a sistemas inteligentes para la toma de decisiones.

En la parte física, representada esquemáticamente en la parte inferior de la Fig. 1, se encuentran interconectadas las diferentes entidades involucradas en el proceso de producción y, en una clasificación general, pueden ser agrupadas como Equipos, Materiales, Operarios y Herramientas. Todas estas entidades pueden compartir información y mantener actualizado su estado en tiempo real gracias a las diferentes tecnologías actuales como, por ejemplo, los protocolos Ethernet, ROS, sistemas de georreferenciación, etiquetas/ lectores RFID, etc. De esta forma, en la parte física no sólo se recopilan e intercambian datos, sino que también se comunica con la parte digital donde se generan decisiones o acciones que posteriormente se ejecutan en la parte física. Es decir, mediante esta comunicación, la parte digital puede afectar los procesos físicos y viceversa.

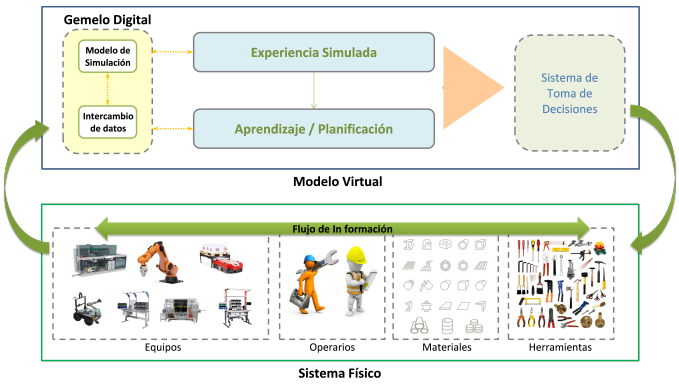


Fig. 1. Jerarquía funcional de entidades en un sistema de manufactura.

La parte digital, representada en el fragmento superior de la Fig. 1, contiene distintos subsistemas, servicios y aplicaciones que desarrollan distintas funciones que permiten a las entidades intervinientes en la fabricación operar coordinadamente para optimizar el desempeño del sistema productivo en su conjunto. Actualmente, muchos campos de investigación enfocan sus esfuerzos hacia el desarrollo de propuestas para este sistema virtual. Particularmente, todos ellos buscan aprovechar las bondades y beneficios de diferentes metodologías para generar propuestas que sustenten el desarrollo de sistemas autónomos de toma de decisiones, los cuales deben establecer las acciones que se deben ejecutar para gestionar el sistema físico de producción [30]. En este sentido, el concepto de GD ocupa un lugar clave y está centrando la atención de muchas investigaciones para el desarrollo de metodologías y técnicas que permitan la creación de manera fiable de dichos GDs [31].

La función principal de los GDs es proporcionar una descripción física, funcional e integral del sistema en su conjunto. Los GDs son modelos virtuales para reproducir de manera realista las propiedades, comportamientos y reglas de las entidades de la parte física [32]. En otras palabras, los modelos virtuales y las entidades físicas tienen una apariencia y comportamientos similares, es decir, son gemelos.

Para el desarrollo de los GDs, las técnicas de simulación tienen un rol central y deben operar en un esquema de intercambio de información con una variedad de datos, provenientes de los diferentes equipos y sistemas físicos de planta, para mantener actualizados continuamente los modelos digitales con el fin de reflejar cualquier cambio que ocurra en los equivalentes físicos en el tiempo. La interacción entre el mundo físico y los GDs tiene una importancia clave debido a que los modelos virtuales y el mundo físico deben evolucionar conjuntamente, ya que este último posee un comportamiento dinámico y, por ejemplo, una misma entidad puede mostrar diversas propiedades en diferentes momentos.

C. Estructura de un GD para Sistemas de Manufactura

Desde una perspectiva jerárquica, los GDs pueden pensarse compuestos por tres niveles diferentes: Entidades, Información y Sistema. La Fig. 2 muestra una composición solapada de los tres niveles mencionados.

En un primer nivel se encuentran los modelos de todas las entidades que intervienen en el sistema físico. En este nivel, cada una de las entidades debe ser modelada digitalmente con la mayor fidelidad posible. Actualmente, existen muchas soluciones y plataformas disponibles empleadas en el ámbito industrial que proveen soluciones para cada una de las entidades. En concreto, para diferentes tipos de entidades se han desarrollado potentes simuladores computacionales que ya han dado cuenta de su eficacia. También, es común que intervengan personas (operarios) para desarrollar tareas específicas, los cuales pueden ser modelados digitalmente, empleando un simulador adecuado [4]. De la misma forma, cuando intervienen entidades que realizan operaciones unitarias pueden ser modeladas mediante entornos de simulación específicos. En definitiva, una vez que se identifican todas las entidades intervinientes en el proceso productivo, cada entidad puede ser comprendida como un subsistema dentro de otro sistema más amplio, y a su vez, existen herramientas computacionales de simulación que permiten crear los modelos digitales más apropiados de cada una de ellas.

Cada uno de los subsistemas modelados no opera de manera aislada, sino que tiene asociado un comportamiento (dinámica) en el que realiza intercambios (por ej. de materiales, energía, insumos, datos, etc.) con otras entidades en el proceso productivo. Es decir, cada una de las entidades presentes tiene un propósito específico y, en consecuencia, tienen asociados un conjunto de funciones con lo cual continuamente están generando datos y/o información que pueden (o no) ser almacenados y/o compartidos entre las distintas entidades físicas y sus modelos digitales.

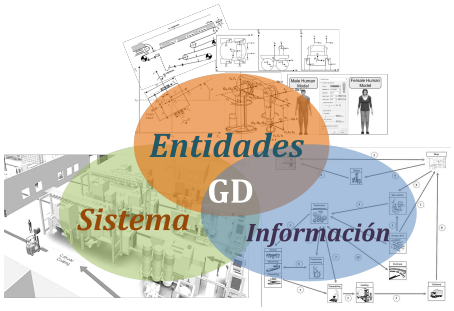


Fig. 2. Relación funcional de un GD para un sistema de manufactura.

En el sistema productivo, las entidades están relacionadas y comparten datos e información, es decir tienen una relación sinérgica y forman parte de una estructura sistémica. Por ello, cada una de estas entidades debe estar adecuadamente modelada y vinculada entre sí cuando se quiere generar el modelo del sistema productivo en su conjunto. En otras palabras, de nada sirve, por ejemplo, modelar perfectamente y de forma aislada un centro de trabajo, un robot manipulador y un operario si no se lo hace con todas las relaciones y propiedades requeridas en el entorno fabril en el que se están empleando.

Dada la heterogeneidad y la complejidad de las entidades que pueden intervenir en la composición de un sistema de manufactura flexible y por lo tanto, para generar los mejores GDs, es conveniente utilizar los simuladores específicos y adecuados para cada entidad acorde a su complejidad y especificidad. En el desarrollo de GDs para sistemas completos de producción manufacturera, la vinculación entre diferentes entornos de simulación y las entidades físicas ha sido escasamente abordada en la literatura. Por ello, en este trabajo se propone una arquitectura de software para integrar diferentes entornos de simulación, adecuados para modelar cada una de las correspondientes entidades comúnmente usadas en los entornos de manufactura flexible, con los dispositivos físicos propiamente dichos (entidades físicas) para dar lugar al GD del sistema que se vinculará con un sistema autónomo de toma de decisiones. Para el efecto de este trabajo, se prueba dicha integración con una planta de referencia vinculada con un sistema autónomo decisor basado en el paradigma de aprendizaje por refuerzos. Esta propuesta se desarrolla en la próxima sección.

IV. Arquitectura de Software Integrada ROS-Tecnomatix

En la actualidad, los sistemas de producción incluyen una multiplicidad y variedad de robots en diversas formas para realizar una plétora de tareas, ya sea para operaciones de transformación como para la manipulación y transporte de materiales y productos en proceso. También es un hecho que la robótica colaborativa ha venido para quedarse y es uno de los pilares de lo que algunos autores ya están llamando como industria 5.0 [33]. Hoy en día, es muy común que los fabricantes de robots y equipos automatizados provean simuladores computacionales de sus productos, los cuales son la base para la creación de los GD de cada equipo (entidad) en cuestión. Sin embargo, muchas de estas soluciones disponibles no ofrecen una solución integral desde la perspectiva del sistema industrial. La diversidad y complejidad de cada una de las entidades que intervienen en un proceso productivo hacen que los simuladores destinados a modelar íntegramente un sistema productivo (por ejemplo los simuladores de eventos discretos), puedan no reflejar con suficiente grado de fidelidad el sistema en su conjunto ya que tienden a simplificar la dinámica de las entidades que lo componen, lo cual va en detrimento de la calidad del GD del sistema industrial que se pretenda crear. Por esto, la integración de un simulador de eventos discretos con las entidades reales junto con sus simuladores especializados, es una opción prometedora para la creación de un verdadero GD de un sistema industrial.

A. ROS en los Sistemas de Manufactura

En el pasado, la interacción y la comunicación con y entre los robots eran complejas y dependientes de la plataforma robótica empleada. Durante la última década, esta situación ha sido facilitada y potenciada debido al creciente desarrollo del Sistema Operativo Robot, conocido como ROS.

ROS consiste en un conjunto de procesos heterogéneos e independientes denominados *nodos* ejecutados al mismo tiempo y teniendo la capacidad de intercambio de mensajes entre ellos. La conexión entre los nodos se denomina *tópico*. Un *nodo* puede publicar mensajes o subscribirse a un determinado *tópico* [34]. ROS permite encapsular nodos desarrollados en diferentes lenguajes de programación, facilitando enormemente el crecimiento incremental del sistema y su prototipado. En efecto, la incorporación de nuevos nodos que manejen nuevos dispositivos, tanto sensores como actuadores, se realiza de forma inmediata y homogénea, aún para dispositivos con drivers realizados en software propietario y muy diferente entre sí. Estas características le confieren a toda la arquitectura una gran versatilidad, fundamentalmente para los trabajos de investigación tecnológica que requieren de una integración rápida y ágil de diferentes componentes de software y hardware.

Resumidamente, las principales características y ventajas de ROS son: (1) abstracción de hardware y control de dispositivos de bajo nivel; (2) intercambio de mensajes entre procesos; (3) gestión de paquetes; (4) herramientas para desarrollar, probar y ejecutar código en varios tipos de lenguajes de programación; (5) computación distribuida; (6) reutilización de software; y (7) pruebas rápidas. En este sentido, todos los robots equipados con ROS comparten el mismo criterio de comunicación.

Para la creación de los GDs, la estandarización de los protocolos de comunicación juega un papel central. Teniendo en mente que los GDs están constituidos por modelos simulados y con fusión de datos reales, es atinado pensar en la creación de estos gemelos considerando la posibilidad de interactuar con las entidades reales mediante el esquema del ROS.

B. Ambiente Simulación de Planta: Software Tecnomatix

Muchos entornos de simulación han surgido desde hace unos años a la fecha. Sin embargo, muchas veces las vinculaciones con los equipos de planta así como la adquisición de

datos en tiempo real suelen ser un obstáculo para los analistas a la hora de pensar en un esquema de intercambio de datos para el desarrollo de GDs. En esta sección presentamos brevemente el paradigma de programación del simulador de procesos de manufactura TPS.

Una particularidad que, en principio, lo vuelve atractivo es que, por una parte, se pueden explorar sus capacidades con licencias de prueba libres y con un número máximo de hasta ochenta entidades, lo cual muchas veces es suficiente para modelar, por ejemplo, un sistema productivo de una pequeña/mediana empresa. Por otra parte, permite la interconexión con software y hardware *in the loop* tipo PLC (*Programable Logic Controller*) y la integración con los productos Siemens es aún más sencilla. En la práctica esto puede ser ventajoso dado que es muy común la existencia de equipamiento compatible en las plantas industriales.

Finalmente otra de las ventajas que brinda el software TPS a la hora de modelar es que tiene una interfaz gráfica amigable que permite modelar un sistema de manufactura simplemente seleccionando cada uno de sus componentes, ubicándolo y estableciendo las relaciones con los demás componentes del sistema. También permite programar lógicas de actuación a cada instancia utilizada con su lenguaje propietario. Asimismo, otra cualidad que lo vuelve aún más atractivo es que cada una de esas instancias permite una conexión directa con Python. Para nuestra propuesta, esto es de suma importancia dado que es posible la conexión con otras instancias cuyo comportamiento sea el del elemento real o su homólogo simulado mediante su específico simulador dinámico. También, otra característica ventajosa de la integración con Python es que facilita la integración con muchas librerías de tratamiento de datos e inteligencia artificial (por nombrar algunas, Pandas, Scikit-Learn, Numpy, Scipy, Pytorch, Tensorflow, Caffe, etc.) y que están en pleno auge en esta era de la Industria 4.0.

C. Integración ROS - Tecnomatix

La Fig. 3, esquematiza la combinación de los elementos individuales descritos en la secciones previas para desarrollar la arquitectura de software propuesta que integra el ROS y el simulador de procesos de manufactura TPS, con la posibilidad del agregado de un sistema de decisión autónoma para el control y optimización de procesos.

En nuestra arquitectura, ROS es considerado como un componente más de la misma y éste se encuentra esquematizado en el sector izquierdo de la Fig. 3. Como se puede observar, este componente se encuentra integrado por dos *nodos*, o procesos heterogéneos e independientes, denominados *roscore* y *Simulación de Planta*. El nodo *roscore* consiste en una colección de procesos y programas del ROS que permite que una red heterogénea de nodos puedan comunicarse entre sí. El nodo *Simulación de Planta* representa el proceso que tiene la responsabilidad de dirigir una planta de manufactura simulada. A su vez, este proceso puede comunicarse con otros *nodos* por medio de *tópicos* del tipo suscriptores o publicadores. Por ejemplo, si la parametrización de las tareas de un proceso productivo (p.e., tiempos, secuencias, etc.) se obtiene mediante un estudio de simulación *off-line*, la misma puede ser especificada a los equipos y componentes de planta intervinientes mediante la publicación directa de las especificaciones a través del tópico correspondiente con las entidades involucradas de la planta real. Por otro lado, el nodo de *Simulación de Planta* puede suscribirse a un tópico de datos proporcionado por una entidad real que impacte en el proceso de simulación de la planta en cuestión. De esta manera, se logra un esquema de realimentación de conocimiento entre el sistema físico y el sistema virtual lo que constituye la base fundamental para un GD de un sistema industrial.

En la parte derecha de la Fig. 3, se encuentra remarcado con una línea de trazo de color celeste el simulador de procesos de manufactura TPS. Con este software es posible diseñar plantas de manufactura adaptadas a cada problemática en particular y capaces de simular sus procesos de producción. Con este software, cada modelo de planta de manufactura se almacena en archivos externos con extensión .spp. Por ejemplo, en la Fig. 4 se muestra el modelo desarrollado en el simulador TPS para un GD de un problema puntual de manejo de materiales en un piso de planta flexible. Este caso de estudio se aborda en detalle en la Sección V.

En la parte central de la Fig. 3, delimitada con una línea de trazo de color verde, se encuentra la integración entre ROS y el software *Tecnomatix Plant Simulation*. Como se puede ver, se propone un patrón de diseño de *clases* siguiendo el paradigma de diseño y programación orientado a objetos para el desarrollo de la arquitectura de software propuesta en este trabajo. El diseño propuesto se implementó y testeó íntegramente utilizando el lenguaje de programación Python. La comunicación entre el ROS y el software TPS se realiza mediante la clase denominada *Interfaz de Conexión*, que tiene la responsabilidad de proporcionar puntos de acceso externos que realizan alteraciones dentro del software de simulación. Estos puntos de acceso comprenden la carga de una planta en particular para ser procesada por el simulador, establecer y obtener datos de una entidad cualquiera simulada, comenzar, parar y pausar simulaciones, entre otras alteraciones permitidas por el software TPS. Para implementar esta comunicación se utilizó la librería *win32com* la cual permite intervenir de forma externa cualquier sistema que sea ejecutado dentro del sistema operativo de Windows®. Particularmente, PyWin32 es una librería escrita en lenguaje de programación Python para el sistema operativo Microsoft Windows, la cual brinda acceso a gran parte de la API de Win32 con la capacidad de creación y utilización dinámica de objetos COM (Component Object Model) permitiendo la comunicación externa entre procesos, en este caso procesos de ROS y Tecnomatix. La clase *Interfaz de Conexión* representa el patrón de diseño Mediator, comúnmente utilizado en el desarrollo de software basado en la programación orientada a objetos, la cual tiene la responsabilidad de encapsular la comunicación entre los objetos [35].

Retomando el diseño de la arquitectura de software propuesta, en la Fig. 3 se puede observar que el nodo *Simulación de Planta* del ROS ejecuta procesos mediante el pasaje de mensaje de una planta en particular utilizando la clase *Planta*. La clase *Planta* representa una clase abstracta y sigue el patrón de diseño denominado herencia simple, la cual abstrae todo el comportamiento general de cualquier planta de manufactura

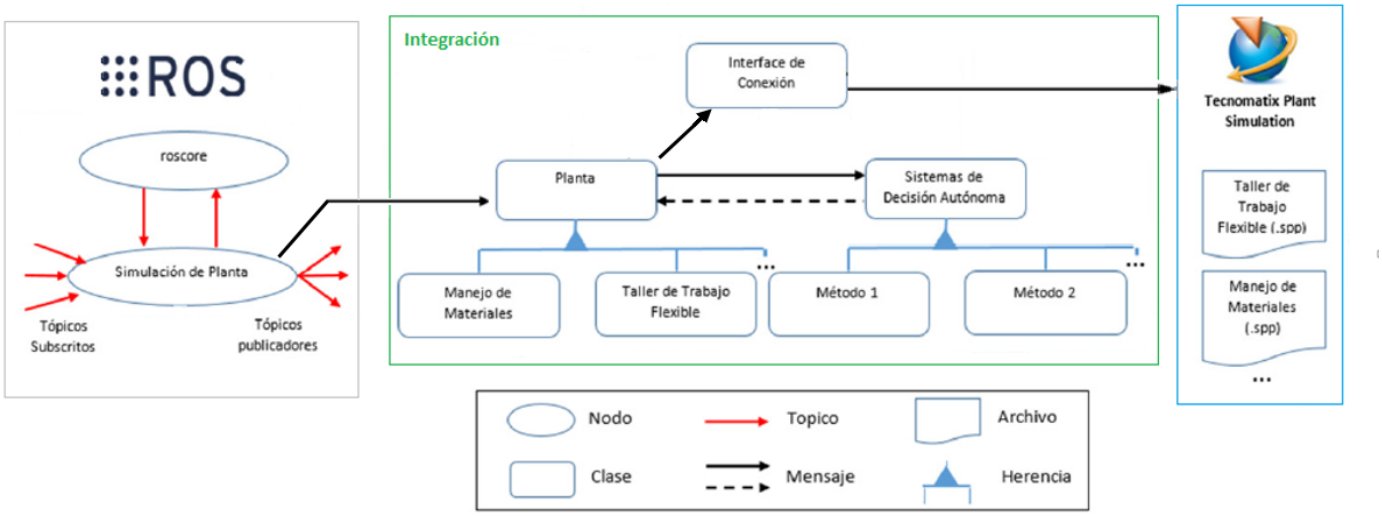


Fig. 3. Arquitectura de software propuesta para integrar ROS y un simulador de procesos de manufactura TPS.

que se desee simular [36]. Precisamente, la clase *Planta* ejecuta procesos mediante el pasaje de mensajes por medio de la clase *Interfaz de Conexión*, la cual representa un comportamiento general que requiere cualquier planta de manufactura simulada coordinada externamente. En esta jerarquía simple se tienen por ejemplo dos clases de plantas de manufactura concretas denominadas *Taller de Trabajo Flexible* y *Manejo de Materiales*, entre otras plantas que pueden ser implementadas. Cada planta concreta implementa el comportamiento de coordinación de una planta simulada por el software de simulación. Por ejemplo, la clase concreta *Manejo de Materiales* tiene la responsabilidad de cargar dentro del software de simulación el archivo Manejo de Materiales con extensión spp, cambiar el comportamiento de la planta simulada ante establecimiento de datos a entidades simuladas, comenzar, parar y extraer resultados de una simulación, siempre a través de los puntos de accesos brindados por la *Interfaz de Conexión*. Se debe tener en cuenta que cada comportamiento específico de cada clase concreta de planta depende de la simulación abordada para cada planta en particular. Asimismo, debe notarse que la extensión de una nueva planta concreta que dirija de forma externa una planta simulada por el software TPS es directa, simplemente se debe crear una nueva clase que herede de la clase *Planta* y codificar el comportamiento simulado requerido, promoviendo la escalabilidad de la arquitectura de software propuesta.

Paralelamente a la jerarquía de planta (ver Fig. 3), se tiene una herencia simple con la clase abstracta *Sistemas de Decisión Autónoma*. Esta clase abstrae todo el comportamiento general de cualquier método de toma de decisiones autónoma basado, por ejemplo, en métodos de inteligencia artificial. Una clase concreta, y como se mostrará en la Sección V, representa la aplicación de un método particular de toma de decisiones (por ejemplo, como en nuestro caso, proponiendo un método de aprendizaje por refuerzo). Notar, que el agregado de un nuevo método de toma de decisiones es directo y simplemente se debe crear una clase que herede de la clase *Sistemas de Decisión Autónoma* y codificar el método en particular, nuevamente fomentando la escalabilidad de la arquitectura de software propuesta.

Estas jerarquías paralelas interrelacionadas entre las clases abstractas *Planta* y *Sistemas de Decisión Autónoma* representa una combinación de patrones de diseño conocidos como Factory Method y Observer o Publish-Suscribe utilizados comúnmente en el desarrollo de software de la programación orientada a objetos [35]. El patrón Factory Method tiene la responsabilidad de construir o fabricar objetos de un subtipo determinado, es decir, plantas simuladas, en la que se le puede o no aplicar dentro de cada subtipo un método en particular, en este caso, un proceso del sistema de decisión autónoma. Por otro lado, el patrón Observer o Publish-Suscribe define una dependencia entre el observador (la clase *Planta*) y observado (la clase *Sistemas de Decisión Autónoma*). De esta manera, una planta cualquiera, es decir, el observador, ejecuta un proceso en particular del sistema de decisión autónoma, es decir, el observado (ver Fig. 3 el paso de mensaje de una flecha de color negra). Luego, cuando el observador cambia de estado, por ejemplo, cuando el método de sistema de decisión autónoma obtiene un resultado en la optimización de un proceso en particular, el observador es notificado automáticamente (ver Fig. 3 el paso de mensaje de una flecha de color negra a trazo), teniendo nuevamente el control de la planta, y a su vez, el resultado del proceso de optimización para que seguidamente altere con el resultado obtenido la planta simulada en cuestión.

Bajo una mirada abstracta de la Fig. 3 podemos notar que desde cualquier nodo del ROS podemos crear numerosas plantas de manufactura a la vez, de igual o de distinto tipo, que interactúen con el simulador de procesos de manufactura TPS, y a su vez a cada tipo de planta se le puede aplicar o no métodos de un sistema de decisión.

D. Sistema Autónomo de TDD Basado en el Gemelo Digital

Como ya hemos señalado, el paradigma de fabricación centrado en el cliente requiere que las empresas se esfuercen

en buscar alternativas para mejorar su competitividad. El hecho de desarrollar sistemas autónomos de toma de decisiones (SATDD), capaces de optimizar productividad global del sistema de fabricación y consecuentemente el uso de recursos, es una ventaja distintiva para la determinación de una variedad de decisiones que deben tomarse constantemente, y en tiempo real, para hacer frente tanto a las variaciones de demanda de clientes, como así también, a las perturbaciones propias de un sistema productivo. Los SATDDs, para ser eficaces, necesitan información precisa y en tiempo real para determinar su política de actuación [7].

En el pasado, los modelos digitales se usaban casi exclusivamente para analizar y validar los comportamientos de los sistemas e instalaciones industriales de producción altamente automatizados. En la actualidad, las tecnologías modernas, la digitalización y la inteligencia artificial fomentan la necesidad de desarrollar aún más las herramientas de planificación digital a la vez que promueven el desarrollo de métodos SATDD.

La propuesta de desarrollo de GDs para sistemas de manufactura, es justamente, uno de los pilares para el desarrollo de SATDD basados en el paradigma del aprendizaje por refuerzos (AR) [37]. En este esquema de AR un agente inteligente (SATDD) interactúa con su entorno (GD) eligiendo acciones cuyo objetivo es maximizar la recompensa esperada. Formalmente, este problema se puede describir como un problema de decisión de Markov [38] el cual requiere que se defina un vector de estado del sistema, el conjunto de acciones posibles a tomar en cada estado y una función de recompensa que premie los efectos de las acciones tomadas. De esta forma, el agente genera su base de conocimiento y adapta su política de actuación de acuerdo a la función de recompensa definida. En la Sección V-B se detalla cada uno de los elementos en base al caso de estudio abordado.

V. Caso de Estudio

En los sistemas de manufactura flexible el manejo de materiales es un problema complejo y que afecta directamente la eficiencia y productividad del sistema de fabricación.

Mejorar la eficiencia de los procesos de fabricación, en particular la logística y el manejo de materiales, es un problema difícil y que ha recibido especial atención, principalmente para la planificación de tareas en esquemas de producción estáticos [39], [40]. Sin embargo, no hay demasiada evidencia para esquemas de producción dinámicos, cuando cambian por ejemplo, alguna de las condiciones de los pisos de planta flexibles y/o también varía la demanda. En esta sección presentamos la creación de un GD para un sistema de manufactura típico de manejo de materiales. La creación del modelo digital en el software TPS y su integración con ROS para el desarrollo de los correspondientes GDs, se realizó mediante la arquitectura presentada en las secciones anteriores. Finalmente, se presentará una propuesta de integración con un sistema de toma de decisiones que emplea un agente artificial simple basado en el AR.

En un problema de manejo de materiales se busca minimizar las distancias recorridas, el uso de los medios de transporte y las demoras en las entregas que interrumpan el sistema productivo. Durante la resolución de un problema de este tipo, se debe evitar la interrupción del flujo de trabajo asegurando la provisión en tiempo y forma de los insumos necesarios para las diferentes estaciones de trabajo procurando lograr el normal funcionamiento del sistema de producción.

A. Definición del Modelo de Simulación

Particularmente, para el análisis se adapta el caso de estudio presentado en el trabajo de [41] el cual presenta un problema de manejo de materiales en piso de planta formulado como un problema de decisión Markoviano para ser abordado mediante el paradigma del aprendizaje por refuerzos para encontrar al menos una política de decisión óptima para el manejo de materiales en planta.

En nuestro caso, el sistema bajo análisis está compuesto por diez estaciones de trabajo ($S_i \forall i = 1, 2, ..., 10$) que consumen simultáneamente dos tipos de parte (P_1, P_2); en cada estación de trabajo existen *buffers* ($B_k \forall k = 1, 2, ..., 20$) de almacenamiento para cada parte cuyas capacidades son iguales al doble de la cantidad de partes de cada tipo consumidas en una hora; hay nueve vehículos de tres tipos diferentes ($V_{je} \forall j \in \{1, 2, 3\} \vee e \in \{1, 2, 3\}$) y se desarrollan distintos planes de manejo de materiales. Cada plan de manejo de materiales tiene asociado una ruta a seguir (W_c), un transporte específico (T_d), partes y cantidades de partes a llevar a las estaciones. Los equipos de transporte no son todos iguales, es decir, cada tipo de vehículo, V_{je}, tiene su propia capacidad de carga y puede llevar cualquier tipo de parte.

La Fig. 4 muestra el modelo desarrollado en el software Tecnomatix. Como se puede observar, en este modelo están presentes todas las entidades que forman el sistema en estudio. Asimismo se aprecian, las vías de circulación con sus respectivos sentidos, los puntos de carga/descarga, los equipos de transporte y las partes.

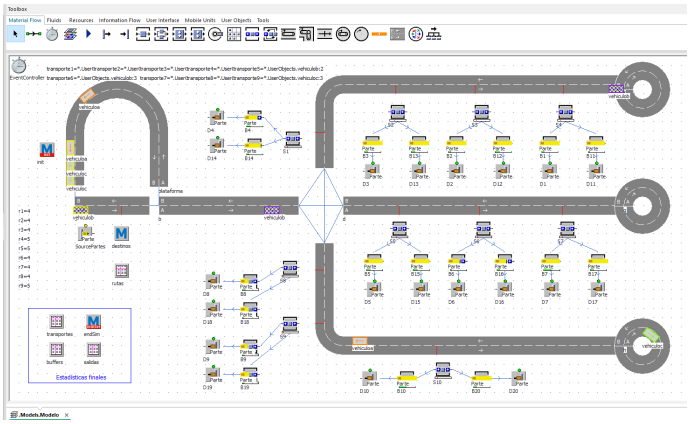


Fig. 4. Modelo de simulación en TPS.

B. Integración Digital y SATDD para el Manejo de Materiales

El modelo en el simulador TPS queda definido mediante un conjunto de parámetros, los cuales pueden accederse externamente mediante el nodo ROS creado. Para poder cambiar los planes se consideran: las distancias recorridas por cada

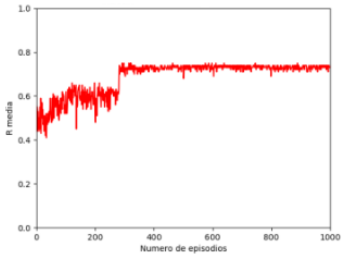
medio de transporte y nivel de ocupación de los mismos, el nivel máximo de llenado de los buffers respecto a la capacidad definida y la porción del tiempo en que se llenan completamente y porcentajes de trabajo y espera en cada salida que consume partes.

Para optimizar el desempeño del sistema la función de recompensa se plantea de forma tal que devuelve un valor +1 sí y sólo sí el plan mejora y 0 de otra forma. Para poder determinar la mejora del plan se evalúa la función $\mathscr{L}(u/s) = C_1w_1 + C_2w_2 + C_3w_3$ en dos momentos sucesivos, y se comparan entre sí. En este caso C_1 es el porcentaje de ocupación de los transportes; C_2 es la porción de tiempo en que los buffers se llenan completamente y C_3 es el porcentaje de tiempo de espera total para consumir las partes. Los factores de peso w_1, w_2 y w_3 se fijaron en 0.3, 0.3 y 0.4, respectivamente. Si el resultado disminuye respecto al anterior, entonces se verifica que el plan ha mejorado.

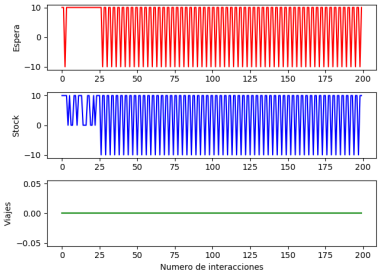
En el módulo de TDD de la arquitectura propuesta, se integró un agente artificial basado en el conocido algoritmo *Q*-learning [37], [42] siguiendo una estrategia de exploración ϵ-*greedy*, con un decaimiento de ϵ con el transcurso de los episodios desde $\epsilon = 1$ hasta un valor mínimo $\epsilon = 0.01$. La tasa de aprendizaje usada es $\alpha = 0.1$ y el factor de descuento es $\gamma = 0.9$. Los resultados obtenidos se muestran en la Fig. 5. La Fig. 5a muestra la recompensa promedio obtenida por el agente durante 1000 episodios de entrenamiento. Cada episodio contempla 100 instancias de interacción. Como puede verse, el agente aprende a mejorar su política de actuación lo que se refleja en un incremento ascendente del valor medio de recompensa. La Fig. 5b muestra las acciones escogidas por el agente en una prueba de testeo de 200 interacciones.

Con el fin de demostrar la capacidad de integración y versatilidad de la arquitectura propuesta, y para realizar una comparación de desempeño con otra metodología de TDD comparable, por otra parte, se implementó en el módulo de TDD un agente artificial basado en el algoritmo SARSA [37] con igual parametrización que en el caso anterior. La Fig. 6 resume los resultados obtenidos para este caso. En la Fig. 6a se muestra la recompensa media obtenida durante el entrenamiento y en la Fig. 6b se muestra un caso de las acciones elegidas en este caso por el agente sistema de toma de decisiones. Como puede inferirse de observar la evolución de la recompensa obtenida, cuando en el módulo de toma de decisiones se implementa un agente basado en el algoritmo *Q*-learning (Fig. 5) se obtienen mejores resultados que cuando se usa un agente basado en el algoritmo SARSA (Fig. 6). Además, esto demuestra la flexibilidad de la arquitectura propuesta para integrar e implementar fácilmente diferentes propuestas de algoritmos para la tomas de decisiones.

Mediante este caso de aplicación simplificado se pudo demostrar cómo usando la arquitectura propuesta se puede integrar de manera muy simple un SATDD, basado en este caso en el algoritmo de aprendizaje *Q*-learning, con un GD de un determinado sistema de manufactura. Estos resultados preliminares, permiten advertir la potencialidad de la arquitectura desarrollada para usarse en el desarrollo de GDs de otros sistemas industriales y que puedan integrarse fácilmente también con otros SATDDs, basados en otras metodologías más complejas como por ejemplo las de aprendizaje profundo (*deep learning*).

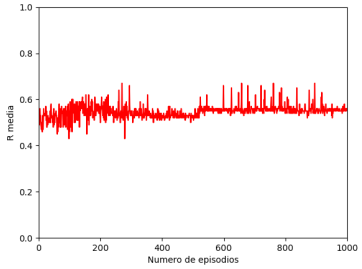


(a) Evolución de la recompensa media por episodio.

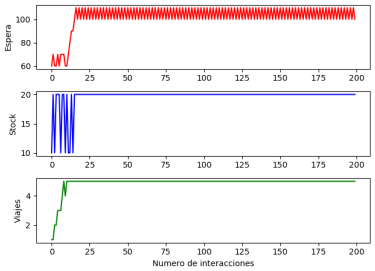


(b) Acciones tomadas por el agente para una prueba de 200 interaciones.

Fig. 5. Resultados del algoritmo *Q*-learning



(a) Evolución de la recompensa media por episodio.



(b) Acciones tomadas por el agente para una prueba de 200 interaciones.

Fig. 6. Resultados del algoritmo SARSA

VI. Conclusión y Trabajo Futuro

El aporte principal de este trabajo es el desarrollo de la arquitectura de software propuesta para la creación de gemelos

digitales mediante la integración del sistema operativo de robots y un simulador de procesos de manufactura flexible. La misma permite la interconexión e interacción a través de intercambio de datos e información entre diferentes componentes (tanto de hardware como de software) que pueden estar presentes en un sistema de fabricación flexible.

Otro aporte singular del trabajo es el desarrollo de una metodología sistemática para la generación de un gemelo digital de un sistema de manufactura y su integración con un sistema autónomo de toma de decisiones. Como consecuencia de esto, también queda establecido un entorno que conectándose naturalmente a través de ROS con la planta de manufactura real, permitirá monitorear y controlar el sistema basándose en el paradigma de GD.

Por otra parte, se usó la arquitectura presentada para desarrollar el gemelo digital de un sistema de manejo de materiales de un piso de planta de un sistema de manufactura particular. Como resultado de esta vinculación, con la arquitectura propuesta se pudo integrar todos los componentes del sistema.

Finalmente el gemelo digital del sistema de manejo de materiales se usó para encontrar una política de toma de decisiones. Para ello, se implementó un sistema de toma de decisiones autónomo basado en un algoritmo de aprendizaje por refuerzos tipo Q-learning para el problema de manejo de materiales en piso de planta. De esta forma, queda demostrada la capacidad de integración de los gemelos digitales de sistemas de manufactura flexible, creados mediante la arquitectura de software propuesta, y su vinculación con un sistema de toma de decisiones.

Si bien, el objetivo principal de este trabajo se centra en lograr una integración entre los distintos componentes y módulos de un sistema de manufactura inteligente, los resultados preliminares obtenidos de la aplicación del algoritmo Q-learning como SATDD son razonables y plausibles de mejoras con lo cual constituyen el punto de partida para futuros desarrollos e integraciones de GDs y SATDDs más sofisticados. En un futuro inmediato se plantea ampliar los desarrollos a casos con mayor cantidad de componentes de hardware y software para abordar otros casos de estudio en vistas a analizar el desempeño de la propuesta realizada.

REFERENCES

[1] A. Kusiak, "Smart manufacturing," *International Journal of Production Research*, vol. 56, no. 1-2, pp. 508–517, jan 2018.

[2] R. Y. Zhong, X. Xu, E. Klotz, and S. T. Newman, "Intelligent Manufacturing in the Context of Industry 4.0: A Review," *Engineering*, vol. 3, no. 5, pp. 616–630, 2017.

[3] I. Zolotová, P. Papcun, E. Kajáti, M. Miškuf, and J. Mocnej, "Computers & Industrial Engineering Smart and cognitive solutions for Operator 4 . 0 : Laboratory H-CPPS case studies," *Computers & Industrial Engineering*, no. xxxx, pp. 1–15, 2018.

[4] S. Baskaran, F. A. Niaki, M. Tomaszewski, J. S. Gill, Y. Chen, Y. Jia, L. Mears, and V. Krovi, "Digital human and robot simulation in automotive assembly using siemens process simulate: A feasibility study," *Procedia Manufacturing*, vol. 34, pp. 986–994, 2019.

[5] C. González García, E. R. Núñez-Valdez, V. García-Díaz, C. Pelayo G-Bustelo, and J. M. Cueva Lovelle, "A Review of Artificial Intelligence in the Internet of Things." *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 5 (Special Issue on Artificial Intelligence Applications), no. 4, pp. 9–20, 2018. [Online]. Available: http://doi.org/10.9781/ijimai.2018.03.004

[6] D. Dejene, B. Tiwari, and V. Tiwari, "TD2SecIoT: Temporal, Data-Driven and Dynamic Network Layer Based Security Architecture for Industrial IoT," *International Journal of Interactive Multimedia and Artificial Intelligence*, pp. 1–11, in Press. [Online]. Available: http://dx.doi.org/10.9781/ijimai.2020.10.002

[7] K. Ding, J. Lei, F. T. Chan, J. Hui, F. Zhang, and Y. Wang, "Hidden Markov model-based autonomous manufacturing task orchestration in smart shop floors," *Robotics and Computer-Integrated Manufacturing*, vol. 61, no. July 2019, pp. 1–9, 2020.

[8] S. Haag and R. Anderl, "Digital twin – Proof of concept," *Manufacturing Letters*, vol. 15, pp. 64–66, 2018.

[9] J. Wang, C. Xu, J. Zhang, J. Bao, and R. Zhong, "A collaborative architecture of the industrial internet platform for manufacturing systems," *Robotics and Computer-Integrated Manufacturing*, vol. 61, no. August 2019, 2020.

[10] J. D. Sterman, *Business dynamics: Systems thinking and modeling for a complex world*. McGraw-Hill Education, 2000.

[11] D. Mourtzis, M. Doukas, and D. Bernidaki, "Simulation in manufacturing: Review and challenges," *Procedia CIRP*, vol. 25, no. C, pp. 213–229, 2014.

[12] N. Furian, M. O'Sullivan, C. Walker, S. Vössner, and D. Neubacher, "A conceptual modeling framework for discrete event simulation using hierarchical control structures," *Simulation Modelling Practice and Theory*, vol. 56, pp. 82–96, 2015.

[13] M. De Paula and E. C. Martínez, "Optimal operation of discretely controlled continuous systems under uncertainty," *Industrial and Engineering Chemistry Research*, vol. 51, no. 42, pp. 13 743–13 764, 2012. [Online]. Available: http://dx.doi.org/10.1021/ie301015z

[14] K. Sofiane and H. Djamila, "A Temporal Distributed Group Decision Support System Based on Multi-Criteria Analysis," *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 5, no. 7, pp. 7–21, 2019. [Online]. Available: http://doi.org/10.9781/ijimai.2019.03.002

[15] F. Amiri, B. Shirazi, and A. Tajdin, "Multi-objective simulation optimization for uncertain resource assignment and job sequence in automated flexible job shop," *Applied Soft Computing Journal*, vol. 75, pp. 190–202, 2019. [Online]. Available: https://doi.org/10.1016/j.asoc.2018.11.015

[16] C. Block, D. Lins, and B. Kuhlenkötter, "Approach for a simulation-based and event-driven production planning and control in decentralized manufacturing execution systems," *Procedia CIRP*, vol. 72, pp. 1351–1356, 2018. [Online]. Available: https://doi.org/10.1016/j.procir.2018.03.204

[17] C. Barrera-Diaz, J. Oscarsson, S. Lidberg, and T. Sellgren, "Discrete Event Simulation Output Data-Handling System in an Automotive Manufacturing Plant," *Procedia Manufacturing*, vol. 25, pp. 23–30, 2018. [Online]. Available: https://doi.org/10.1016/j.promfg.2018.06.053

[18] F. Z. Ben Moussa, R. De Guio, S. Dubois, I. Rasovska, and R. Benmoussa, "Study of an innovative method based on complementarity between ARIZ, lean management and discrete event simulation for solving warehousing problems," *Computers and Industrial Engineering*, vol. 132, no. March 2018, pp. 124–140, 2019. [Online]. Available: https://doi.org/10.1016/j.cie.2019.04.024

[19] M. H. Mourad, A. Nassehi, D. Schaefer, and S. T. Newman, "Assessment of interoperability in cloud manufacturing," *Robotics and Computer-Integrated Manufacturing*, vol. 61, no. June 2018, p. 101832, 2020. [Online]. Available: https://doi.org/10.1016/j.rcim.2019.101832

[20] Y. Lu, C. Liu, K. I. Wang, H. Huang, and X. Xu, "Digital Twin-driven smart manufacturing: Connotation, reference model, applications and research issues," *Robotics and Computer-Integrated Manufacturing*, vol. 61, p. 101837, 2020. [Online]. Available: https://doi.org/10.1016/j.rcim.2019.101837

[21] J. Cheng, H. Zhang, F. Tao, and C. F. Juang, "DT-II:Digital twin enhanced Industrial Internet reference framework towards smart manufacturing," *Robotics and Computer-Integrated Manufacturing*, vol. 62, no. January 2019, p. 101881, 2020.

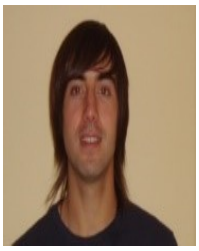
[22] O. Meski, F. Belkadi, F. Laroche, and B. Furet, "Towards a knowledge-based framework for digital chain monitoring within the industry 4.0 paradigm," *Procedia CIRP*, vol. 84, pp. 118–123, 2019. [Online]. Available: https://doi.org/10.1016/j.procir.2019.04.250

[23] K. Zhang, T. Qu, D. Zhou, H. Jiang, Y. Lin, P. Li, H. Guo, Y. Liu, C. Li, and G. Q. Huang, "Digital twin-based opti-state control method for a synchronized production operation system," *Robotics and Computer-Integrated Manufacturing*, vol. 63, no. November 2019, p. 101892, 2020. [Online]. Available: https://doi.org/10.1016/j.rcim.2019.101892

[24] B. Illmer and M. Vielhaber, "Synchronizing digital process twins between virtual products and resources – A virtual design method,"

Procedia CIRP, vol. 84, pp. 532–537, 2019. [Online]. Available: https://doi.org/10.1016/j.procir.2019.04.227
[25] C. Qian, Y. Zhang, C. Jiang, S. Pan, and Y. Rong, "A real-time data-driven collaborative mechanism in fixed-position assembly systems for smart manufacturing," *Robotics and Computer-Integrated Manufacturing*, vol. 61, no. July 2019, 2020.
[26] J. Leng, Q. Liu, S. Ye, J. Jing, Y. Wang, C. Zhang, D. Zhang, and X. Chen, "Digital twin-driven rapid reconfiguration of the automated manufacturing system via an open architecture model," *Robotics and Computer-Integrated Manufacturing*, vol. 63, no. November 2019, 2020.
[27] R. P. Rolle, V. D. O. Martucci, and E. P. Godoy, "Architecture for Digital Twin implementation focusing on Industry 4.0," *IEEE Latin America Transactions*, vol. 18, no. 5, pp. 889–898, 2020.
[28] C. G. Cassandras and S. Lafortune, *Introduction to discrete event systems*. Springer Science+Business Media, 2008.
[29] Q. Qi, F. Tao, T. Hu, N. Anwer, A. Liu, Y. Wei, L. Wang, and A. Y. Nee, "Enabling technologies and tools for digital twin," *Journal of Manufacturing Systems*, no. August, pp. 0–1, 2019. [Online]. Available: https://doi.org/10.1016/j.jmsy.2019.10.001
[30] M. Kunath and H. Winkler, "Integrating the Digital Twin of the manufacturing system into a decision support system for improving the order management process," *Procedia CIRP*, vol. 72, pp. 225–231, 2018. [Online]. Available: https://doi.org/10.1016/j.procir.2018.03.192
[31] X. Sun, J. Bao, J. Li, Y. Zhang, S. Liu, and B. Zhou, "A digital twin-driven approach for the assembly-commissioning of high precision products," *Robotics and Computer-Integrated Manufacturing*, vol. 61, 2020.
[32] R. He, G. Chen, C. Dong, S. Sun, and X. Shen, "Data-driven digital twin technology for optimized control in process systems," *ISA Transactions*, vol. 95, pp. 221–234, 2019.
[33] K. A. Demir, G. Döven, and B. Sezen, "Industry 5.0 and Human-Robot Co-working," *Procedia Computer Science*, vol. 158, no. November, pp. 688–695, 2019. [Online]. Available: https://doi.org/10.1016/j.procs.2019.09.104
[34] A. Koubaa, *Robot Operating System (ROS) The Complete Reference*. Springer,1st ed, 2016.
[35] M. Shaw and D. Garlan, *Software architecture: perspectives on an emerging discipline*. Upper Saddle River, NJ, 1996.
[36] L. Bass, P. Clements, and K. R., *Software architecture in practice*. Addison-Wesley Professional; 3rd Edición, 2013.
[37] A. G. Sutton, R. S., & Barto, *Reinforcement learning: An introduction*. MIT Press, 1998.
[38] G. E. Monahan, "State of the art - A survey of partially observable Markov Decision Processes: theory, models, and algorithms," *Management Science*, vol. 28, no. 1, pp. 1–16, jan 1982. [Online]. Available: http://mansci.journal.informs.org/content/28/1/1.full.pdf+html
[39] S. Jun, S. Lee, and Y. Yih, "Pickup and delivery problem with recharging for material handling systems utilising autonomous mobile robots," *European Journal of Operational Research*, no. In Press, 2020.
[40] A. Gola and G. Kłosowski, "Development of computer-controlled material handling model by means of fuzzy logic and genetic algorithms," *Neurocomputing*, vol. 338, pp. 381–392, 2019. [Online]. Available: https://doi.org/10.1016/j.neucom.2018.05.125
[41] S. Govindaiah and M. D. Pey, "Applying reinforcement learning to plan manufacturing material handling Part 1: Background and formal problem specification," *ACMSE 2019 - Proceedings of the 2019 ACM Southeast Conference*, pp. 168–171, 2019.
[42] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, no. 8, pp. 279–292, 1992. [Online]. Available: https://doi.org/10.1007/BF00992698

Carolina Saavedra Sueldo nació en Olavarría, Buenos Aires, Argentina en 1989. Se graduó como Ingeniera Industrial en 2014, por la Universidad Nacional del Centro de la Pcia. de Buenos Aires y desde el año 2019 es estudiante del Doctorado en Ingeniería de la misma Universidad. Como becaria doctoral de la CICpBA, sus investigaciones se centran en tecnologías de la Industria 4.0 combinando el uso de técnicas de simulación e inteligencia artificial para el desarrollo de gemelos digitales.



Sebastian A. Villar graduado de Ingeniero en Sistemas en la Facultad de Ciencias Exactas de la Universidad Nacional Centro de la Provincia de Buenos Aires (UNCPBA), Argentina (2009), y de la Maestría en Administración de Empresas (MBA) en la Facultad de Ciencias Económicas de la UNCPBA (2011), y doctor en Ingeniería en la Facultad de Ingeniería de la UNCPBA (2014). También investigador asistente de la Comisión de Investigaciones Científicas de la Provincia de Buenos Aires (CICBA), trabajando en el Grupo de Ingeniería INTELYMEC del Centro de Investigaciones en Física e Ingeniería del Centro de la Provincia de Buenos Aires (CIFICEN). Tema principal de interés procesamiento de imágenes Robótica, SONAR.

Mariano De Paula es Ingeniero Industrial, graduado en la Facultad de Ingeniería de la Universidad Nacional del Centro de la Provincia de Buenos Aires (UNCPBA), Argentina (2007). Se ha graduado como Doctor en Ingeniería en la Universidad Tecnológica Nacional (UTN-FRSF), Argentina (2013). Es investigador del Concejo Nacional de Investigaciones Científicas y Técnicas de Argentina (CONICET) y desempeña su actividad en el grupo INTELYMEC, UNCPBA. Además, se desempeña como profesor Adjunto en la Facultad de Ingeniería de la UNCPBA.

Gerardo G. Acosta es Ingeniero en Electrónica por la Universidad Nacional de La Plata, Argentina (1988) y Doctor en Informática por la Universidad de Valladolid, España (1995). Es Investigador Independiente del Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), Argentina, Director del Grupo de Investigación y Desarrollo "INTELYMEC", en la FI-UNCPBA e integrante del Consejo Directivo del CIFICEN (UNCPBA – CICPBA – CONICET). Sus intereses de trabajo comprenden el uso de la inteligencia computacional en el control automático, particularmente las técnicas de control inteligente en robótica, terrestre y subacuática. Cuenta con más de ciento ochenta publicaciones y dos registros de propiedad intelectual en éste y otros temas relacionados. Es Senior Member del IEEE desde 2001 y forma parte del Administrative Committee de la OES de IEEE por el período 2015-2016 y 2018-2023. Forma parte de Ad Hoc Strategic Planning Committee de dicha sociedad y es editor asociado en temas oceánicos de la revista EARTHZINE patrocinada por IEEE OES.