Universitat Autònoma de Barcelona

Cómo utilizar simuladores basados en multiagentes para scada en aplicaciones del ámbito industrial

Ismael Fabricio Chaile Alfaro

ADVERTIMENT. L'accés als continguts d'aquesta tesi queda condicionat a l'acceptació de les condicions d'ús establertes per la següent llicència Creative Commons: http://cat.creativecommons.org/?page_id=184

ADVERTENCIA. El acceso a los contenidos de esta tesis queda condicionado a la aceptación de las condiciones de uso establecidas por la siguiente licencia Creative Commons: http://es.creativecommons.org/blog/licencias/

WARNING. The access to the contents of this doctoral thesis it is limited to the acceptance of the use conditions set by the following Creative Commons license: https://creativecommons.org/licenses/?lang=en

ESCOLA D'ENGINYERIA

DEPARTAMENT DE MICROELECTRÒNICA I SISTEMES ELECTRÒNICS

CÓMO UTILIZAR SIMULADORES BASADOS EN MULTIAGENTES PARA SCADA EN APLICACIONES DEL ÁMBITO INDUSTRIAL

por

Ismael Fabricio Chaile Alfaro

MEMORIA DEL TRABAJO DE TESIS DEL PROGRAMA DE DOCTORADO EN MICROELECTRÓNICA Y SISTEMAS ELECTRÓNICOS.

LÍNEA DE INVESTIGACIÓN: SISTEMAS ENCASTADOS INDUSTRIALES, SISTEMAS MULTI ROBOT Y TECNOLOGÍAS DE SISTEMAS DISTRIBUÍDOS EMERGENTES

BELLATERRA (BARCELONA), ESPAÑA

DICIEMBRE 2015

UNIVERSITAT AUTÒNOMA DE BARCELONA

Departament de Microelectrònica i Sistemes Electrònics

Doctorat en Microelectrònica i Sistemes Electrònics

CÓMO UTILIZAR SIMULADORES BASADOS EN MULTIAGENTES PARA SCADA EN APLICACIONES DEL ÁMBITO INDUSTRIAL

Memoria del trabajo de tesis para obtener el grado de Doctor en Microelectrónica y Sistemas Electrónicos.

Autor: Ismael Fabricio Chaile Alfaro

Director: Prof. Lluís Ribas Xirgo

Dr. Lluís Ribas Xirgo, Profesor de la Universitat Autònoma de Barcelona

CERTIFICA:

que la presente memoria titulada "**Cómo utilizar Simuladores Basados en Multiagentes para SCADA en Aplicaciones del Ámbito Industrial**", presentada por Ismael Fabricio Chaile Alfaro para obtener el grado de Doctor en Microelectrónica y Sistemas Electrónicos, ha sido realizada bajo su dirección en la Universitat Autònoma de Barcelona.

Prof. Lluís Ribas Xirgo

Bellaterra (Barcelona), 14 de Diciembre de 2015

Este trabajo de investigación ha sido financiado por la Universitat Autònoma de Barcelona a través de una beca de post graduado bajo el concepto de Personal Investigador en Formación (PIF).

A mi familia,

“Keep yourself in holistic peace. Do always the best you can. Do it with love.”

Anonymous.

Resumen

Esta tesis aborda el problema de cómo utilizar simuladores ABM para la supervisión, control y adquisición de datos (SCADA) en aplicaciones del área industrial. Mayormente el uso de los simuladores se limita a la etapa de diseño y en algunos casos se extiende a la interacción con el entorno real de manera temporal, como es el caso de HIL o Co-Sim. En este trabajo se propone su uso desde la etapa inicial, hasta el despliegue del mismo, dejando la simulación para SCADA permanente de la aplicación final.

Implementar la solución de esta manera tiene, al menos, una ventaja: la posibilidad de agilizar el despliegue de sistemas que interactúan con la realidad, como es el caso del entorno industrial. Esto se puede lograr por medio de una serie de herramientas que se presentan en esta tesis y que permiten aprovechar gran parte del modelo que ya ha sido verificado y validado mediante simulación y embeber únicamente una pequeña parte del mismo en el componente de la realidad que se desea supervisar y/o controlar. Realizar SCADA en tiempo real utilizando simuladores, como se plantea en esta tesis, presenta un principal desafío, el de poder cerrar el lazo de supervisión y control lo suficientemente rápido para que no haya pérdida de mensajes y datos. Para hacer frente a este desafío, se dota a los componentes de la realidad de una cierta autonomía, convirtiéndolos en agentes y se realiza una representación del entorno real mediante una simulación que se sincroniza con la realidad, todo ello concebido mediante una entidad denominada ALOHA.

ALOHA significa *Abstraction*/*Architecture*/*Approach that Leads to Organize Hybrid*/*Heterogeneous Agents* y, aunque el nombre representa un acrónimo, parte de una base más filosófica en la que se considera que todo agente 'educado' debería saludar al iniciar y finalizar el acto comunicacional.

Utilizar simuladores con el enfoque planteado en esta tesis implica proveer al diseñador de un entorno de trabajo (*framework*) integrado por herramientas que permitan implementar el modelo del sistema, la arquitectura de control, el agente ALOHA, el modelo de sincronización y una metodología que sugiera cómo llevar a cabo todo el proceso desde el diseño hasta el despliegue.

Implementar lo ya detallado dentro del entorno industrial implica también que esta solución tenga en cuenta los desafíos actuales y futuros de esta área, como pueden ser cuestiones relacionadas con robustez, flexibilidad, adaptabilidad, evolutividad etc., sin perder de vista los requerimientos de las tendencias a nivel industrial como lo son la Industria 4.0, las *Smart Factories* y el I2oT (de *Industrial Internet of Things*).

Finalmente, para la validar la hipótesis de esta tesis, se construyeron dos demostradores en

dos aspectos distintos del control en el área industrial, el primero relacionado con el sistema de transporte de muestras en laboratorios automáticos utilizando AGVs y el segundo, con el control de un elemento de un sistema de manufactura, representado por una celda de producción compuesta por un brazo neumático. Los dos sistemas han sido desarrollados e implementados con las herramientas elaboradas fruto de este trabajo y los resultados, analizados en cuanto a calidad de sincronización, restricciones de tiempo real y agilidad en cuanto a su implementación.

Abstract

This thesis addresses the problem of how to use ABM simulators for supervisory control and data acquisition (SCADA) of applications in the industrial area. Typically, simulators are used in the design processes of SCADA developments and, in some cases, they require interaction with the real environment temporarily, like when HIL or Co-Sim techniques are used. This work proposes the use of simulators from the initial stage to the deployment phase, keeping them as SCADAs for the final applications.

To implement the solution as proposed in this work has, at least, the advantage of shortening the deployment stage of systems that interact with reality, such as the industrial ones. This can be achieved through a series of tools presented in this thesis and that take advantage of as much of the simulation model as possible, i.e. only a small part of the very model that has been verified and validated during simulation is required to be embedded into the real components that will be monitored and/or controlled. To perform real-time SCADA by using simulators, as proposed in this thesis, presents a major challenge, defined as the need of closing the control loop quickly enough to not losing any message. To face this challenge, the components of reality are provided by a certain autonomy, thereby, turning them into agents and, representation of real environment is shown by performing a simulation that is synchronized with reality, all conceived by an entity called ALOHA.

ALOHA stands for *Abstraction/Architecture/Approach that Leads to Organize Hybrid/Heterogeneous Agents* and, although the name is an acronym, it has a philosophical basis which considers that every 'educated' agent should greet at the start and end of the communicational act (as most of humans do).

Simulators' use under the proposed approach involves providing the designer of a working environment (framework) made of tools to implement the system model, the control architecture, the ALOHA agent, the synchronization module, and a methodology to carry the whole process out from early design stages to final deployment phases.

The creation of the above mentioned framework in the industrial domain cannot be done without taking into account the current and future challenges in this area, such as issues related

to robustness, flexibility, adaptability, scalability, evolvability, etc., without losing sight of the requirements of the industry trends, as presented in Industry 4.0, Smart Factories and I2oT (Industrial Internet of Things), for instance.

The hypothesis of this work has been validated by building two distinct demonstrators in the industrial control domain, the first one related to the sample transport system in automated laboratories by using AGVs, and the second one, to a production cell that uses a pneumatic robot-arm. Both systems have been developed and implemented with the tools of this work and results have been analyzed for synchronization quality, real-time constraints and implementation agility.

Agradecimientos

Resulta de gran satisfacción para mi poder expresar mi gratitud, a todas las personas que formaron/forman parte de mi vida, desde que nací y en especial aquellas que han estado durante estos cinco años, ya sea de una manera u otra, quienes, considero que han contribuido a que este trabajo pueda ser llevado a cabo.

Cuando se comienza una tesis, existen numerosos factores que pueden llevar a su no consecución. Siendo un gran privilegio, el poder llegar a su depósito y posterior defensa. Considero que es algo concedido, pero también fruto del trabajo, la perseverancia, la aptitud y la actitud, como así también, de que se hayan producido los encuentros adecuados a lo largo de la vida. Me refiero con esto último a lo más importante de este mundo, las personas, con su diversidad, impredecibilidad y capacidad de dar amor asociada.

En primer lugar, deseo agradecer a mi mentor, Lluís, quien con su buena fe, excelencia, generosidad, amistad, perseverancia e innumerables otras buenas cualidades, me ha guiado/ ayudado a transitar este proceso tan interesante y de crecimiento tanto profesional como personal. Quisiera también extender este agradecimiento a su familia, la que gentilmente ha cedido muchas veces un tiempo muy valioso. ¡Muchas gracias Lluís y familia!.

Quiero también agradecer a la Universitat Autònoma de Barcelona por haber apostado en mí y a la comisión evaluadora que siempre de manera asertiva ha formulado comentarios y consejos que han ayudado a mejorar este trabajo. ¡Muchas gracias!.

Deseo también expresar mi gratitud a Jose Barata y su equipo de Lisboa, por haberme recibido y permitido ser uno más entre ellos, durante mi Estancia de Doctorado. ¡Muchas gracias!.

A Filippo deseo agradecer por la gran calidad y dedicación en cuanto a las correcciones tanto del inglés así como la redacción de mis artículos. ¡Muchas gracias Fil!.

Me ilusiona expresar mi eterna gratitud a Ana y Luis, quienes han sido benefactores de esta tesis y quienes me han permitido tener una mejor calidad de vida, por medio de su generosidad y confianza. ¡Muchas gracias Ana y Luis!.

No puedo dejar de expresar mi gratitud a dos queridos amigos, Gustavo Sbrugnera y Sergio Nuñez, a quienes conocí hace más de 20 años y, son los que despertaron en mí, la pasión por

la programación, la electrónica, la robótica y todo lo asociado a ello; ambos, siempre compartieron y comparten conocimientos y consejos con gran generosidad. ¡Muchas gracias Gus y Sergio!.

También quiero agradecer a Bruno, un hermano de la vida; a Iris, mi gran amor; a Violeta y Casandra, mis dos princesas, quienes involuntariamente, han resignado compartir sus momentos importantes con su tío; a Joaquín, mi hermano y amigo, además de un gran ejemplo para mí, a Tati, un puente esencial hacia mis sobrinas y, por supuesto, a mi madre, abuela y abuelo, a quienes admiro. ¡Muchas gracias!.

He dejado como penúltimo párrafo, a mi familia, quién está (desde antes que tenga uso de razón) y sé que siempre estará, apoyándome, alentándome, guiándome y alegrándose por mis logros, ya que estos también son vuestros. ¡Muchas gracias!.

Por último, me gustaría expresar también mi gratitud con la siguiente nube de palabras.

Tabla de contenidos

Lista de figuras

Listado de acrónimos

ABM	Agent Based Model/Modelling
ALOHA	Abstraction/Approach/Architecture that Leads to Organize Hybrid/Heterogeneous Agents
SYM	Synchronization Middleware/Module
FIPA	Foundation for Intelligent Physical Agents
ACL	Agent Communication Language
SCADA	Supervision Control and Data Acquisition
AGV	Automated Guided Vehicle
HMI	Human Machine Interface
MCL	Main Control Loop
RTU	Remote Terminal Unit
HIL	Hardware in-the-loop
CO-SIM	Co-Simulation
LIMS	Laboratory Information Management System
WCET	Worst Case Execution Time
FSM	Finite-state machine
FSM	Flexible Manufacturing System
ISM	Immediately Synchronization Method
EDM	Event Discovery Method
MAS	Multi-Agent System
DAI	Distributed Artificial Intelligence
ACM	Association for Computing Machinery
INCOSE	International Council of Systems Engineering

1 Introducción

1.1 Problema objeto de investigación

Los simuladores son programas específicos que permiten implementar y ejecutar modelos de diferente naturaleza, pudiendo ser estos, elementos de la realidad o bien modelos teóricos, como por ejemplo pruebas de conceptos. Al ejecutar la simulación lo que se busca es verificar y validar el/los modelos bajo estudio. Estos modelos y sus interrelaciones son analizados normalmente bajo diferentes escenarios que constituirán el dominio de validez de los mismos [1].

Los comienzos de las simulaciones por ordenador tienen su origen alrededor del año 1947 [2], con la invención del método Monte Carlo y el comienzo de la era de los ordenadores [3]; sin embargo, unos cincuenta años antes, se encuentra lo que es considerada la primera simulación realizada manualmente, la cual, se usó para verificar la forma exacta de probabilidad de la función de densidad *Student's t-distribution* [4].

Durante casi 30 años, hasta llegar a la década del 70, los programas de simulación fueron evolucionando lentamente, aumentando en disponibilidad de programas tanto genéricos como específicos e incrementando su uso principalmente en la comunidad científica. Entre la década de los 80 y 90, hasta nuestros tiempos, las simulaciones comenzaron su auge a pasos agigantados, pasando a extenderse de manera masiva a diferentes ramas de la ingeniería, las ciencias sociales y hacia los ordenadores personales. Dado que los simuladores se han caracterizado desde su aparición por ser muy demandantes de recursos de cómputo (o bien cuando aparecieron la capacidad computacional no estaba a la altura…) y, uno de los factores que podría haber impulsado el citado auge, es el crecimiento de la capacidad de cómputo de los ordenadores, sumado a que la relación precio/capacidad de cómputo fue reduciéndose con los años [5], [6], lo que permitió que los ordenadores pudieran extenderse a los hogares y no estuvieran limitados a Universidades, Gobiernos, Empresas, etc., como sucedía en los inicios.

A medida que los simuladores evolucionaban y popularizaban, también lo hacían los lenguajes de programación y los paradigmas asociados a los mismos. En la actualidad, existen más de

8500 lenguajes de programación [7] siendo algunos mono-paradigma y otros multi-paradigma. Los paradigmas varían desde los más concretos, como pueden ser los imperativos, funcionales, etc., hasta aquellos con niveles superiores de abstracción, donde encontramos los orientados a objetos, a holones y a agentes. Los lenguajes de programación, al igual que los paradigmas, son herramientas que se utilizan en los simuladores.

A medida que crece la abstracción del paradigma, es posible afrontar aplicaciones más complejas, mediante razonamientos más cercanos al dominio del problema y, el hecho que los agentes sean entidades autónomas activas más abstractas que los objetos (normalmente considerados pasivos), sumado al principio de racionalidad de los mismos, los hace más idóneos para dominios concretos de aplicación [8], como por ejemplo, los del ámbito industrial.

El ámbito industrial ha avanzado a pasos agigantados desde la primera revolución industrial hasta la actualidad (tercera revolución industrial). Las revoluciones se caracterizaron por procesos de cambios significativos e incorporación de nuevas tecnologías, como por ejemplo, la máquina a vapor, la energía eléctrica, el motor de combustión interna, la introducción y mejora de los medios de transporte, las comunicaciones, la producción masiva de bienes manufacturados, la automatización, entre otras [9]–[11].

En la actualidad, y más aún en el futuro, con la denominada cuarta revolución industrial, Industria 4.0 o *Smart-factories* [12], [13], los sistemas de manufactura y transporte tanto interno/externo, entre otros, necesitan de tecnologías que ayuden a volverlas más eficientes, adaptables, flexibles, escalables, robustas (tolerante a fallos), sostenibles (que puedan perpetuarse como negocio aún ante cambios significativos del entorno), y en definitiva más competitivas para afrontar los requerimientos actuales y futuros del mercado [14], [15]. Parece lógico pensar bajo este contexto, que cuanto más rápido/ágil pueda ser el proceso que va desde el modelado/diseño, hasta el despliegue de sistemas industriales, esto podría llevar a reducir costes (optimizar uso de recursos tales como dinero, tiempo, etc.) no recurrentes asociados a dicho proceso. Ejemplos de ello podrían ser la implantación de una nueva fábrica, la producción *ad-hoc* de un nuevo producto y, porque no, aquellos sistemas relacionados con la manufactura y el transporte.

A lo largo de los años, el sector industrial ha comenzado a adoptar gradualmente conceptos y modelos con tendencia a las *Smart-factories/Virtual-factories*. Sin embargo, estas tecnologías y en especial aquellas relacionadas con la Supervisión el Control y la Adquisición de Datos (del inglés SCADA) están todavía en su infancia, son caras de implementar, complejas y difíciles de llevar a cabo. Es por ello que las investigaciones deberían moverse en la dirección de SCADA en tiempo real a nivel industrial [5].

Afortunadamente, tanto la tecnología a nivel de capacidad de cómputo así como el *software* [16] (*frameworks*, simuladores, etc.) y los paradigmas asociados a estos han evolucionado lo suficiente, como para permitir que la tecnología basada en agentes, se incorpore gradualmente al área industrial [17]–[21], como solución a algunos de los problemas planteados anteriormente. Sin embargo, a pesar que la tecnología de agentes se comenzó a estudiar en 1980 y no hasta mediados de 1990 comenzó a ser reconocida ampliamente [22], ésta, todavía se encuentra en una etapa emergente, por lo menos en lo que respecta a su aplicación a nivel industrial, donde todavía se necesitan más casos de éxito que muestren tanto sus ventajas, aplicabilidad y sirvan para ganar la confianza del sector en esta tecnología.

Con todo, en esta tesis se plantea un enfoque que permitirá utilizar los simuladores basados en modelos de agentes (del inglés ABM) para SCADA de aplicaciones del dominio industrial, intentando de ésta manera hacer frente a los desafíos planteados anteriormente y, en especial, estudiar las ventajas de dicho enfoque, en cuanto a la agilización de las etapas que van desde el diseños hasta la implementación de sistemas industriales e, intentar de esta manera, validar dicho enfoque para al menos un rango de aplicaciones de este entorno.

A través de la Fig. 1 se presenta una visión general de la tesis, donde en la parte inferior se muestra el enfoque a utilizar y en la parte superior una rápida comparación con soluciones actuales del estado del arte, en cuanto al desarrollo y despliegue de sistemas SCADAS.

La intención de la Fig. 1 es la de explicar fácilmente lo que se desarrollará a la vez de comparar con otras soluciones actuales.

En la parte superior de la Fig. 1, se muestra el entorno virtual (V_E), correspondiente a la simulación. En la parte inferior de la misma, el *shop-floor* representa el entorno real (R_E). R_E está compuesto por diferentes componentes (CO_N) que pueden ser, diferentes clases de robots

(AGVs, brazos robots, etc.), Unidades Remotas (RTU), etc. El modelo de los componentes (*CO*) vive en V_E como V_{CO}, por ejemplo V_{CO1} correspondería al modelo del componente CO_1.

Para construir el V_E con sus correspondientes V_{CO}, se necesita un proceso de transformación que toma las especificaciones de los *CO* de R_E. Esta situación se puede presentar al momento de modelar una planta ya existente, o bien cuando se registran parámetros nominales de especificaciones de los *CO* con el objetivo de realizar un nuevo diseño.

Una vez que el R_E esta virtualizado, sus modelos (varios paradigmas se pueden utilizar para implementarlos) pueden ser verificados y validados (V & V). Luego de ello, un proceso inverso de transformación comienza desde V_E hacia R_E, conocido como implementación o despliegue que acontece en R_E.

Si el objetivo de la transformación es el de realizar un SCADA permanente de R_E, usualmente es implementado mediante *Frameworks* con diferentes niveles de abstracción, tales como los de Orientación a Objetos (OO), Orientación a Agentes (AO) u Orientación a Holones (HO), los cuales pueden ser de naturaleza genérica (Qt, JADE, JACK™, JANUS) o bien específicos (UNICOS, CORFU, etc.). Bajo esta circunstancia, nuevamente se tienen que realizar la V & V a nivel del *Framework* (aún cuando se haya realizado generación automática de código) y de R_E.

En este contexto, cuando el V_E se ejecuta al ritmo de su tiempo virtual (*VT*), y R_E se ejecuta en tiempo real (*RT*), existe una brecha entre ellos, debido a que no existe una conexión temporal entre los mismos.

En el caso de realizar un SCADA de manera temporal, como se suele hacer para plataformas de *tests*, como ser simulaciones hardware-in-the-loop (HIL) o Co-Simulaciones (CO-SIM), en este caso existe una conexión temporal entre V_E y R_E y muchas veces también entre los correspondiente *VT* y *RT*. La intencionalidad de este escenario es la de conducir a procesos de V & V más rápidos y precisos y cercanos a la realidad así como el de agilizar el despliegue en R_E. Pero, dado que el SCADA de R_E en estas situaciones no es permanente, muchas veces debido a restricciones de tiempo real, seguridad, no disponibilidad o accesibilidad al *CO* del R_E, etc., se utilizan otras soluciones (por ejemplo *Frameworks ad-hoc*) para el control supervisión y adquisición de datos de manera permanente en el componente final.

En la parte inferior de la Fig. 1 se presenta el objetivo de esta tesis, la propuesta de utilizar simulaciones ABM como SCADA de R_E. Para ello, V_E se convierte en un entorno virtual de agentes (VA_E), donde A_E (círculo con rayas) representa a infraestructuras genéricas del entorno virtual, lugar donde los agentes de los *n* componentes A_{COn} viven e interactúan y, donde a su vez, también existen otros agentes llamados Agentes de Recursos Específicos de Aplicación (A_{ASR}). De manera genérica, los A_{CO} están organizados en un alto nivel, o deliberativo (*H*) y en un bajo nivel o reactivo (*L*). *H* es utilizado para las comunicaciones de alto nivel entre los A_{CO} y también con los A_{ASR}, durante la etapa de simulación, así como cuando el sistema simulado interactúa con la realidad. Cuando la simulación se ejecuta en concurrencia con el entorno real (R_E), los componentes del *shop-floor* se comunican a través de *H*. *L* interactúa con sensores y actuadores reales/virtuales de R_E/A_E respectivamente. En R_E, el *L* es llamado L_R y en VA_E, el *L* es llamado L_V. Hasta aquí, se ha presentado la nomenclatura a nivel general y básica de esta tesis.

La particularización de los componentes convertidos en agentes viene denotada al agregar un subíndice que varía de 1 a *n*. A modo de ejemplo, una interacción entre el agente A_{CO1} con A_{Con} se realizaría en el alto nivel entre H_1 y H_n, que también puede ser escrito $H_{1,n}$, a su vez, internamente H_1 interactúa bidireccionalmente a través de SYM (módulo de sincronización de la simulación con la realidad) con su correspondiente bajo nivel L_{V1} y L_{R1}. Respectivamente, cada L_{V1} y L_{R1} interactúan con sus correspondientes entornos A_E y R_E. Este ejemplo es también análogo para cualquier A_{CO}. Por el momento se ha presentado la nomenclatura esencial pero no todavía la entidad principal, ALOHA.

ALOHA significa *Abstraction/Approach/Architecture that Leads to Organize Hybrid/Heterogeneous Agents*. Los agentes híbridos/heterogéneos son en este sentido del tipo ALOHA, como se puede observar en el interior del cono invertido truncado. La heterogeneidad emerge de la naturaleza de sus entidades dado que ALOHA vive en ambos mundos (R_E y V_E) y en ninguno al mismo tiempo. En este sentido ALOHA es considerado/nombrado a lo largo de esta tesis como entidad/arquitectura/enfoque/agente, aunque también se le podría llamar holon. El origen de ALOHA no tan solo nace de las palabras que integran el acrónimo, sino que el autor de esta tesis considera que desde el punto de vista comunicacional, unos agentes 'educados' deberían saludarse tanto al iniciar como al

finalizar la comunicación. Por último pero no menos, ALOHA tiene una tercera entidad, a la cual se la llama SYM (*Synchronization Middleware/Module*). SYM es un recurso virtual (*software*, pero no limitado a ello) que ayuda con la interacción/concurrencia de ambos mundos/entornos (virtual y real), entre otras cosas.

Durante la etapa de diseño en el simulador y luego en la etapa de despliegue, hay procesos de verificación y validación (V & V). Para el caso del entorno simulado, verificar y validar VA_E implica V & V H, L_V, A_E y A_{ASR}. Sin embargo, una de las ventajas del enfoque ALOHA, es que cuando VA_E ya está verificado y validado, únicamente L_V es transformador/trasladado/recodificado/embebido/adaptado en L_R, lo cual implica que únicamente esta parte necesita ser V & V. Esto podría llevar a agilizar el despliegue del sistema así como su verificación y validación, entre quizás muchas otras ventajas.

Con todo, el agente/entidad ALOHA es implementado siguiendo una metodología/guía para el enfoque ALOHA, la cual se expone en el Capítulo 4.

Presentada ALOHA, dentro del contexto general planteado en la Fig. 1 y con el objetivo de mostrar con claridad en qué se diferencia esta tesis con otras soluciones encontradas en el estado del arte, donde también se presentan simulaciones que interactúan con la realidad, se pueden nombrar algunos de los trabajos más cercanos en este sentido y que están basados en HIL y CO-SIM [23]–[44] cuya descripción de algunos de ellos se cita en el Capítulo 2. La diferencia de esta tesis con los citados trabajos radica en que en ninguno se utilizó la misma tecnología *software*, ninguno de ellos está basado en ABM y, aunque en algunos de ellos existe el concepto de sincronización, no ha sido desarrollado ni presentado como un modelo genérico y modular, trasladable a otra soluciones y embebido a nivel de agente, como lo es en esta tesis; al mismo tiempo, ninguno de ellos ha sido desarrollado a través de un enfoque ALOHA y tampoco en ninguno se plantea dejar una simulación ABM como SCADA de la realidad de manera permanente.

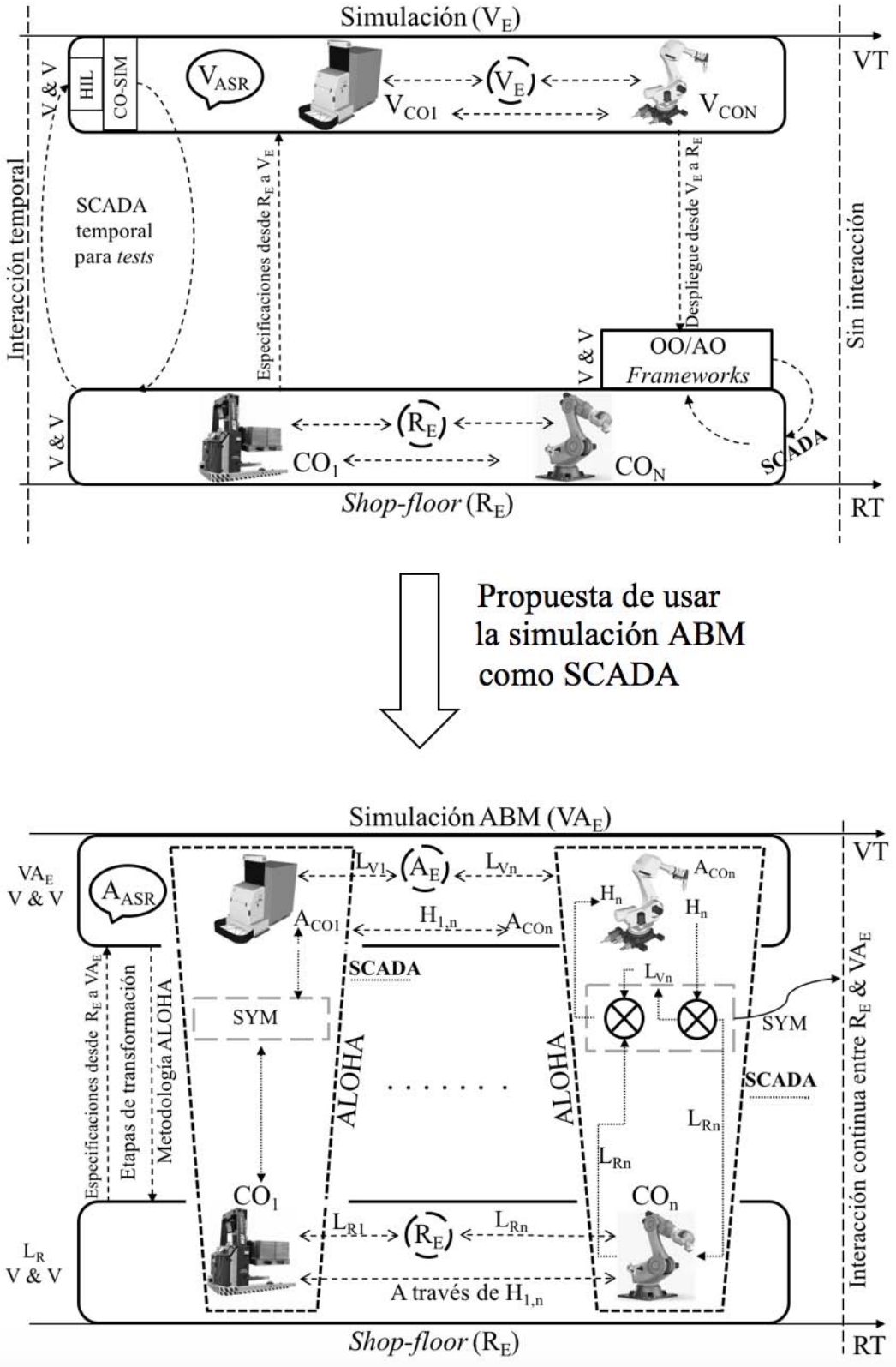


Fig. 1 Vista general del trabajo de Tesis

Es por ello, que para el mejor conocimiento del autor de esta tesis, el trabajo presentado en este documento se podría considerar original.

A continuación se detallan los aspectos científicos investigados en el marco de esta tesis.

La pregunta de investigación a contestar durante este trabajo es:

¿Es posible utilizar simuladores ABM desde la etapa de diseño hasta el despliegue para SCADA en tiempo real de aplicaciones industriales?

La hipótesis a validar de este trabajo de investigación es:

Como consecuencia de mantener el simulador ABM desde el diseño hasta el despliegue, se podrían reducir costes no recurrentes de ingeniería, dado que el enfoque permitiría agilizar la implantación de sistemas SCADAs.

Por lo tanto, el enfoque seguido para hacer frente a la pregunta de investigación ha sido el siguiente:

- Los sistemas multi-agentes parecen ser un paradigma de modelado e implementación adecuado para proveer a las soluciones industriales de aspectos tales como flexibilidad, robustez, adaptabilidad, escalabilidad, etc.
- Los simuladores ABM son una herramienta *software* normalmente utilizada en la etapa de diseño, para verificación y validación de modelos en un entorno de cierta complejidad, los cuales podrían extenderse a la etapa de despliegue en aplicaciones industriales.
- Podrían existir aplicaciones del dominio industrial donde se pueda realizar SCADA en tiempo real desde un simulador ABM.
- Los componentes del entorno industrial a controlar/supervisar con este enfoque tendrán que ser adaptados/modelados/transformados para convertirse en partes del agente que los representa en el sistema.
- Es posible que se necesiten crear unas series de herramientas (arquitecturas, modelos, módulos específicos, etc.) que permitan llevar a cabo la pregunta de investigación.

Las principales contribuciones científicas desarrolladas en el marco de esta tesis son:

1. Proveer de un marco/conjunto de herramientas para mostrar cómo utilizar simuladores ABM para SCADA en tiempo real del dominio industrial.
2. Presentar el enfoque/arquitectura/agente ALOHA, que permite integrar el componente de la realidad como agente con entidades virtuales y reales sincronizadas.
3. El módulo de sincronización, interno a ALOHA, que permite la concurrencia entre el mundo virtual y el real.
4. Una metodología simple que sugiere cómo llevar a cabo sistemas basados en agentes, en donde se utilizan simuladores durante todo el proceso, desde el diseño hasta la solución final, y que además integra agentes tipo ALOHA.

Con todo, la idea alrededor de esta tesis es la de aprovechar que el/los modelos de la realidad que ya se encuentran verificados y validados en el simulador puedan utilizarse para SCADA de la realidad, trasladando únicamente una parte del código hacia el componente ubicado en el *shop-floor*.

A continuación se detalla cómo se organizada esta tesis.

1.2 Esquema de la tesis

Esta tesis está organizada en seis capítulos: Introducción, Estado del arte y trabajos relacionados, Marco conceptual, Herramientas utilizadas en el enfoque ALOHA, Prototipos y validación en el área industrial, Conclusiones y trabajo futuro.

La **Introducción** es el capítulo actual, sirve para motivar al lector respecto al dominio bajo análisis, muestra una visión general de soluciones actuales en el estado del arte junto a la propuesta de esta tesis, el problema de investigación y el enfoque adoptado para el desarrollo de este trabajo. Al final de este capítulo se presenta el esquema de la tesis.

En el capítulo 2 (**Estado del arte y trabajos relacionados**), se presentan diferentes trabajos (tesis, artículos publicados en revistas, proyectos de investigación, etc.) que están relacionados con esta tesis, algunos que pueden compararse de manera más directa, como pueden ser las simulaciones y co-simulaciones, las aplicaciones de agentes en el área industrial, diferentes arquitecturas de agentes, sistemas de agentes en tiempo real, etc. En este capítulo algunos trabajos se comparan siguiendo una estructura de un breve resumen seguido de las diferencias

y similitudes con esta tesis, mientras que otros simplemente se nombran para mostrar puntos de coincidencias y puntos de divergencia. Hacia el final del capítulo se identifican y analizan diferentes tendencias, problemas en común y se sugieren diferentes sub–áreas de investigación que pueden resultar de interés dentro del contexto de esta tesis.

En el capítulo 3 (**Marco conceptual**), se muestran brevemente los principales conceptos utilizados para el desarrollo de esta tesis, con ello se pretende proveer al lector de un marco teórico que facilite la comprensión de este trabajo. Es por ello que en la sección 3.1 se describen brevemente conceptos de paradigmas con niveles de abstracción tales como el de los objetos, holones y agentes; en la sección 3.2 el enfoque esta puesto en dos tecnologías diferentes que permiten implementar programas para ordenadores, como lo son los simuladores y los *frameworks*. En este capítulo, el mayor énfasis se pone en los simuladores, por ser la tecnología utilizada en este trabajo y en los agentes, por ser el paradigma base de esta tesis.

El capítulo 4 (**Herramientas utilizadas en el enfoque ALOHA**), presenta las principales herramientas que han sido creadas y luego utilizadas durante esta tesis. Entre las herramientas se encuentran cuatro entidades bien diferenciadas, estando la sección 4.1 enfocada en el desarrollo del modelo basado en ABM y la arquitectura de control para aplicaciones industriales basadas en agentes que implementan una arquitectura tipo ALOHA para representar agentes de este tipo; la sección 4.2 muestra la arquitectura interna de lo que es considerado un agente tipo ALOHA con los diferentes niveles y módulos por los que está compuesto; en la sección 4.3 se presenta el módulo de sincronización, que es una capa intermedia del agente tipo ALOHA y cuya función es la de sincronizar la representación virtual del componente de la realidad que está siendo modelado; finalmente en la sección 4.4 se crea una metodología/guía/diagrama de flujo, que pretende sugerir cómo implementar paso a paso el *framework* (que incluye agentes tipo ALOHA) desarrollado en el marco de esta tesis.

El capítulo 5 (**Prototipos y validación en el área industrial**), está integrado por dos aplicaciones del área industrial donde se utilizan las herramientas desarrolladas en esta tesis para su implementación. La primera, representa un *testbed* donde se utilizan AGVs para el transporte de muestras en una aplicación de laboratorio de análisis clínicos. La segunda,

representa una celda de producción que utiliza un brazo robot neumático para mover objetos entre dos puntos específicos. Ambos demostradores han sido desarrollados siguiendo la metodología ALOHA propuesta y poseen agentes tipo ALOHA entre los componentes del sistema. Los resultados presentados en este capítulo muestran la viabilidad de: utilizar simuladores para SCADA en tiempo real de aplicaciones industriales; usar la misma herramienta de simulación durante todo el proceso de diseño, desde su modelado hasta el despliegue; sincronizar la simulación basada en agentes con la realidad y por consiguiente algunos beneficios asociados a ello, como pueden ser, pronósticos más fiables, posibilidad de diagnóstico de fallos, balance entre supervisión y sincronización, etc. Finalmente, como resultado también se desprende, que esta propuesta permitiría agilizar/acelerar la implementación global de sistemas industriales complejos.

Finalmente, en el capítulo 6 (**Conclusiones y trabajo futuro**), se presentan las conclusiones principales de este trabajo, se resumen los principales resultados y contribuciones, a la vez, se proponen sintéticamente posibles direcciones hacia donde canalizar trabajos futuros, que podrían ser de interés para el estado del arte o futuras tesis.

2 Estado del arte y trabajos relacionados

En este capítulo se citan diferentes publicaciones y tesis doctorales, las cuales se consideran relevantes dentro del marco de este trabajo. Se organiza por medio de un pequeño resumen del artículo/tesis seguido de una comparación con el trabajo de esta tesis. Luego son nombradas de manera más breve algunas diferencias específicas y puntos de coincidencia con respecto a otros trabajos. Al final de este capítulo, se realiza un análisis más profundo, se identifican tendencias y problemas comunes así como sus consecuencias.

A continuación, se realiza un amplio y profundo análisis, en donde se muestran trabajos interesantes en simulaciones (basadas y no basadas en agentes), aplicaciones industriales y no industriales de modelado basado en agentes (ABM), donde alguno de ellos incorporan características de tiempo real y sincronización.

En [45] se describe la simulación de una aplicación industrial de producción de pintura, basada en un diseño distribuido de MAS para la ejecución de sistemas de manufactura denominados (@MES). El sistema está basado en unidades autónomas que llenan la brecha existente entre la planificación de la producción y el control a nivel de *shop-floor*. Este artículo propone una metodología orientada a agentes para el diseño y verificación de sistemas basados en @MES. En cada unidad autónoma, tanto para las órdenes o los recursos, el agente juega un rol de gestor a través de la implementación del *scheduling* y las funciones de control tanto para las órdenes y los recursos. El diseño propuesto en @MES favorece a la generación de un *scheduling* emergente y la ejecución de un control local que provee mayor agilidad y sensibilidad. Con este fin, un detallado mecanismo de interacción es modelado y simulado usando Netlogo, con el objetivo de garantizar que, mientras los agentes de órdenes y recursos ejecutan sus funciones y juegan su rol, las propiedades emergentes de @MES verifican que la solución es estable y con la capacidad de rechazar de manera robusta las perturbaciones. Con este trabajo, los puntos en común son la utilización de Netlogo para implementar el ABM para una aplicación del área industrial, el uso de MAS y la verificación y validación por medio de simulación del sistema representado, diferenciándose esta tesis principalmente por la extensión de la simulación para SCADA de la planta real.

En [46] se describe un caso de estudio basado en la logística externa de camiones. Está basado en MAS para la gestión de las cargas y el control de los camiones. El programa ha sido desarrollado en JACK™ y permite a los usuarios realizar simulaciones basadas en diferentes escenarios para su análisis así como la planificación de las cargas en tiempo real. En este contexto, se utiliza una arquitectura basada en agentes denominada *belief-desire-intention* (BDI) con el objetivo de modelar de manera dinámica las cargas y la planificación de los camiones. Se consideran las dinámicas reales de los sistemas de transporte dado que la mayoría de los algoritmos para el dominio del transporte disponibles en la literatura están generalmente enfocados en la generación de soluciones efectivas ya sea para problemas de *planning*/*scheduling*. El trabajo de esta tesis se parece con [46] respecto a la aplicación industrial de sistemas de transporte y la consideración de la simulación en tiempo real basada en MAS. Diferenciándose en que en esta tesis se contempla que se puede realizar SCADA de la planta real desde la simulación conjuntamente con la sincronización de simulación con realidad.

En [47], se introduce un detallado análisis de las etapas más significativas en la evolución de las tecnologías de simulación, incluyendo recientes aplicaciones a nivel industrial y de investigación en un período que va desde la década del 70 hasta el 2014. Esta revisión identifica brechas en las prácticas actuales, tendencias a futuro y desafíos en este campo. El artículo propone que las empresas están abiertas a adoptar estos conceptos y los modelos en lo que es considerado las fábricas virtuales (VF). Sin embargo, las tecnologías relacionadas a VF, y en especial estas que involucran a sistemas SCADA están todavía en su infancia, caracterizándose por un coste alto y una implementación compleja. Por estas razones, los investigadores deberían inclinarse a crear más demostradores que muestren la viabilidad de el control y el monitoreo en tiempo real de fábricas además de desarrollar más herramientas que muestren aplicabilidad y asequibilidad de estos sistemas. Por ende, se propone que se debe trabajar más en ello desarrollando herramientas más inteligentes, las que conducen a sistemas más autónomos y auto adaptables. Se concluye que aún si existe un desarrollo significativo de nuevas herramientas de simulación, esto es sin duda un campo fértil para la investigación. En este sentido el autor de esta tesis está de acuerdo con la mayoría de lo citado y en consecuencia, esta tesis busca contribuir con su granito de arena en la dirección indicada en este artículo.

SimIShop, un prototipo para entornos de simulación flexibles y basado en es descripto en [48]. El mismo puede ser adaptado para resolver los nuevos requerimientos a nivel de fabricación y es usado para validar la arquitectura propuesta, que está basada en agentes, para la simulación de sistemas de fabricación inteligentes. Las herramientas utilizadas para su implementación han sido JADA, MySQL y PROTEGÉ-OWL. El prototipo provee flexibilidad y proactividad con el fin de interactuar y utilizar recursos provistos por el modelo y la información generada durante el proceso de simulación. El prototipo también es escalable, lo que permite al usuario escalar la herramienta según sus necesidades. Durante la fase de creación, la herramienta provee lo que es llamado un *Synchronization Manager*, que se encarga de supervisar la creación de los modelos. La meta de SimIShop es la creación de un entorno real de un sistema de fabricación dentro de un entorno simulado; teniendo en cuenta requerimientos específicos relacionados con la "*new manufacturing era*", tales como organización distribuida, interoperabilidad, cooperación, escalabilidad, tolerancia a fallos y agilidad, la mayoría no contempladas por un gran número de herramientas de simulación. El punto en común con este trabajo es en el uso del paradigma de agentes y la simulación de entornos industriales que tienen en cuenta los requerimientos actuales del área industrial, sin embargo, ambos trabajos se basan en diferentes tecnologías *software*. Otra de las diferencias es que en esta tesis el trabajo incluye la interacción de la simulación con la planta real en tiempo real y sincronizada con la misma, mientras que en [48] la solución aportada es a nivel únicamente software.

Un estudio basado en simulaciones que pone un énfasis particular en el manejo de sistemas de manufactura bajo condiciones dinámicas es presentado en [49]. En particular, considera sistemas donde las estaciones de trabajo y elementos de transporte son conectados y desconectados en cualquier momento en una topología dinámica, gestionando órdenes de naturaleza aleatoria. Bajo este contexto dinámico y complejo de interacciones en el *shop-floor* hay una necesidad de *rescheduling* con cierta frecuencia tanto para las órdenes actualmente en proceso como de aquellas por ingresar. Por lo tanto, hay un entrelazamiento entre la gestión del material y las actividades de *scheduling*. Para estudiar estos escenarios, se proponer una arquitectura heterárquica basada en el paradigma de MAS. La arquitectura contempla diferentes formas de transporte y provee la característica de ser *plug and produce* así como la capacidad de adaptación a las dinámicas de la producción. Para validad la propuesta, un *testbed*

basado en sistemas de transporte por medio de cintas transportadoras, y otros elemento tales como estaciones, compuertas y puntos de entrada y salida han sido emulados utilizando una interfaz de simulación. La tecnología utilizada para la implementación se basó en un dominio simulado y un dominio basado en agentes programados en JADE y que son FIPA compatibles. La interfaz entre los dominios ha sido implementada por una librería que integra el control de bajo nivel programada *ad-hoc*, la cual se asegura de que todos los elementos de transporte son capaces de interactuar con dispositivos físicos o simulados. Los resultados sugieren que la arquitectura propuesta puede hacer frente a condiciones muy altas de dinamismo tales como cambios topológicos en tiempo de ejecución. Los puntos en común de esta tesis con [49], es que ambos se basan en MAS y utilizan un dominio simulado para aplicaciones industriales así como el uso de elementos de transporte que son capaces de interactuar tanto con dispositivos simulados y reales. La diferencia de esta tesis es que se implementa la sincronización de la simulación con el entorno real, así como la posibilidad de crear entornos de realidad mixta. Otras diferencias son respecto a que la arquitectura utilizada y las herramientas *software*. La meta del trabajo presentado en [49] es la de desarrollar una arquitectura capaz de encarar condiciones de extremo dinamismo y cambios de topologías, mientras que en esta tesis la meta consiste en mostrar que desde una simulación ABM sincronizada con la realidad se puede realizar SCADA en tiempo real y que gracias a la arquitectura y metodologías planteadas esta idea puede llevar a acortar las etapas entre el diseño y el despliegue de sistemas industriales.

En [50], [51] se presenta un modelo holónico basado en dos puntos de vista: uno que considera los sistemas de transporte y el otro a los edificios hacia donde estos se dirigen. El trabajo describe un *framework* genérico para el modelado y análisis de sistemas complejos naturalmente distribuidos basados en un paradigma del tipo *Holonic Multi-Agent System* (HMAS). La idoneidad del modelo genérico se ilustra a través de la simulación de la planta de producción de automóviles PSA Sochaux con unas dimensiones mayores a 250 hectáreas y una producción diaria de más de 1700 automóviles y con alrededor de 1600 camiones ingresando a la planta cada día. Hechos que llevan a que la planta esté en perpetua evolución. La implementación se realizó utilizando un patrón de diseño clásico como lo es el *Model-View-Controller* (MVC), para desconectar lo que son los datos de la manera en la que la información es representada y manejada desde la vista del usuario. El enfoque está basado en

la formación dinámica de Holones, esto significa que los agentes encuentran y eligen dinámicamente el Holon con el cual mezclarse, resultando de esta manera una holarquía que reproduce la topología de la planta. Los vehículos se mueven en el nivel más bajo de la holarquía, interactuando principalmente con puntos de unión y de intercambio. El *framework* ha sido formalizado utilizando la metodología RIO para probar propiedades importantes concernientes a la capacidad de auto organización de los modelos. La implementación de los modelos se realizó usando Madkit al cual se le realizaron modificaciones *ad-hoc* e incorporaron extensiones para emular la característica holónica de los agentes y manejar algunas carencias de Madkit. El HMAS descripto es utilizado para simular el tráfico dentro de la planta y para identificar los edificios caracterizados por un alto tráfico de camiones. Este enfoque permite analizar y comprender el impacto entre actividades dependientes, mientras que otros simuladores se concentra únicamente en el tráfico y su análisis. Este trabajo y la tesis aquí presentada comparten el hecho de utilizar un paradigma de un alto nivel de abstracción en el área industrial, particularmente en aplicaciones de logística. Respecto a las holarquías formadas por los modelos de [50], [51], las mismas permiten reproducir la topología de la planta mediante formaciones dinámicas de Holones, mientras que en esta tesis no hay formaciones dinámicas de holarquías ya que los modelos y arquitecturas presentados se enfocan únicamente en el sistema de transporte. Sin embargo, el trabajo en [50], [51] está limitado únicamente a simulaciones, mientras que en esta tesis la simulación es usada para dirigir e interactuar con la planta real, entre otras cosas.

En [52] se propone un *toolkit* para ayudar a los diseñadores a construir de manera más rápida aplicaciones basada en MAS, al cual le llaman Multi-Agent Modeling Toolkit (MAMT). Este *toolkit* ayuda a crear un entorno de desarrollo para agentes, donde los desarrolladores pueden modificar diferentes parámetros relacionados con la reusabilidad agentes ya desarrollados y existentes así como la personalización de agentes ya construidos, etc. MAMT provee a los diseñadores con una serie de *templates* de agentes basados en el estándar FIPA, los cuales han sido desarrollados utilizando JADE. A través de estos recursos, los diseñadores pueden construir aplicaciones basadas en MAS con mayor rapidez, haciendo de esta manera la vida de los desarrolladores más fácil y ahorrando tiempo, gracias a estos agentes ya desarrollados. Los agentes pueden ser agregados al área de dibujo arrastrándolos hacia ella, construyendo de

esta manera modelos AUML que usan notación AUML. Luego de ello es posible convertir el AUML en código Java con un solo *click*. Los nuevos agentes pueden ser agregados al repositorio de agentes o bien los agentes existentes se pueden modificar según los requerimientos de la aplicación. El *toolkit* propuesto puede ser utilizado para diseñar/modelar/simular aplicaciones de redes basadas en MAS, aplicaciones distribuidas, aplicaciones basadas en agentes móviles, etc. MAMT ofrece las siguientes funcionalidades: un modelo para *Multi-Agent Rapid Application Development* (RAD), reusabilidad de agentes ya desarrollados, el modelado de los agentes usando AUML, un repositorio de agentes, modificación y verificación del comportamiento de los agentes, conversión de AUML a código funcional Java, siendo la mayoría de estas funcionalidades no incluidas en otras herramientas/metodologías basadas en agentes. En esta tesis se comparte con MAMT, el desafío de proveer con una herramienta que permita a los diseñadores construir aplicaciones basadas en MAS que incluyan una herramienta adecuada para el diseño, modelado y simulación, sin embargo, en esta tesis también se agrega la posibilidad de extender la simulación para que interactúe con el sistema modelado (la realidad), la sincronización entre ambos mundos y la posibilidad de entornos de realidad mixta.

El artículo presentado en [53] describe el estado del arte hasta 2007 para aplicaciones industriales basadas en las tecnologías de agentes. El estudio se ha focalizado en tres áreas: el control holonico, la gestión de la producción y las empresas virtuales. El trabajo presenta siete áreas a nivel industrial que son de relevancia para las técnicas orientadas a agentes, de las cuales cuatro están altamente relacionadas con el trabajo de esta tesis: Control en tiempo real flexible para entornos de fabricación discretos, control inteligente de sistemas altamente distribuidos, el transporte y manejo de materiales y la gestión de la producción. El trabajo presentado en [53] concluye que todavía hay un largo camino antes que la visión basada en agentes y las tecnologías MAS sean adoptadas ampliamente por la industria. En consecuencia, los autores proponen algunos *tips* a tener en cuenta con el objetivo de promover la adopción de esta tecnología. En este sentido, el autor de esta tesis está de acuerdo y cree que se necesitan aún más soluciones basadas en MAS con la visión de incrementar la confianza del área industrial en estas tecnologías. En esta misma línea, este trabajo de tesis intenta contribuir con un *framework* y *testbeds* basadas en tecnología MAS para el área industrial, buscando mostrar

las ventajas de esta tecnología con el fin de intentar masificar la adopción de la misma en las industrias.

En [54] se presenta una plataforma que integra simulaciones con sistemas SCADA a nivel industrial el cual soporta acceso a datos complejos y la reutilización del código de simulación. La plataforma soporta la integración de simulaciones y sistemas SCADA para simplificar el acceso a simulaciones desde la HMIs. El *framework* está basado en una arquitectura de dos niveles: una encargada del nivel técnico y la otra para la semántica de la aplicación, la cual no se encarga de ninguna cuestión técnica, como por ejemplo proveer soporte de configuración al nivel técnico, el diseño de modelos para simulación, etc. El artículo demuestra a través de dos casos la factibilidad de que los modelos simulados dan un soporte eficiente a lo largo de todo el ciclo de vida de sistemas de automatización y control. Los beneficios de la metodología propuesta consisten en la reducción de tiempo y costes de desarrollo y despliegue, ayuda a mejorar cuestiones de seguridad de las soluciones y hacen el rediseño más flexible permitiendo también la reutilización de modelos de simulación así como de otras herramientas y conocimiento a nivel industrial. El punto en común de esta tesis con este trabajo es que ambos utilizan la simulación y los modelos implementados en ella para SCADA industrial, también ambos trabajos proponen la reutilización del código de simulación buscando reducir tiempo y costes de desarrollo y despliegue. Sin embargo, ambas soluciones se implementan con diferentes paradigmas, siendo el de esta tesis el de orientación a agentes mientras que [54] se basa en orientación a objetos, perdiéndose con ello las ventajas que presenta el primero respecto a este último. Otro aspecto diferencial es que en esta tesis se incorpora la sincronización del modelo implementado en el simulador con la realidad y la capacidad de proveer entornos de realidad mixta.

En [55] se presenta NIDOS (*Newspaper Industry Distribution Optimization Software*) el cual está basado en MAS. Este programa es usado para hacer frente a tareas de planificación en uno de los periódicos más grandes de Alemania; en este tipo de industria, de hecho, la sincronización entre la producción y la distribución se está convirtiendo en algo cada vez más importante, con el objetivo de minimizar costes y maximizar la satisfacción de los clientes. NIDOS está basado en JADE, MySQL-Server e Hibernate para *O/R-mapping*. La solución basada en MAS utiliza varios agentes de edición y de transporte. Los agentes de transporte

tienen dos intenciones: una es intentar suplir todos los puntos de entrega a tiempo y maximizar su nivel de servicio (así como el nivel global también), la otra intención es minimizar los costes de transporte mediante la optimización de las rutas. Un agente de coordinación llamado AntTabu es el responsable de los pasos de pre optimización, eso significa construir una ruta óptima para el vehículo teniendo en cuenta los pronósticos de producción provistos por el departamento de edición. Como resultado, durante la implementación de este sistema, se muestra que una optimización del estilo centralizada de este problema conduce a resultados aún peores comparados con la actual solución basada en MAS. Los resultados demuestran que la variable relacionada con costes disminuye alrededor de un 16%, la satisfacción de los clientes se incrementa en alrededor del 1% en promedio y que el número de vehículos a utilizar puede ser reducido en alrededor del 18%. Los puntos de coincidencia de esta tesis con este trabajo es que ambos se basan en MAS y las aplicaciones estudiadas están relacionadas con la logística del transporte, con el objetivo de mejorar/optimizar resultados de negocio. Ambas soluciones también utilizan el concepto de sincronización pero de diferente manera, en [55] entre dos etapas diferentes del negocio mientras que en esta tesis entre la simulación y la realidad con el objetivo de obtener mejores pronósticos y en consecuencia en ambos casos mejorar los resultados del negocio y la toma de decisiones. Ambos trabajos difieren en que utilizan diferentes tecnologías para implementar la solución en el artículo se basa en un *framework* como lo es JADE, mientras que en esta tesis se utiliza un simulador ABM (Netlogo). Otra diferencia entre ambos trabajos es que [55] no es una solución basada en SCADA mientras que esta tesis contempla la SCADA desde el simulador.

iCoSim-FSM, un prototipo funcional de control adaptativo inteligente basado en un simulador donde los modelos se implementan con el paradigma de orientación a objetos es presentado en [56]. El prototipo ofrece un control de la planificación de la producción a nivel integral, a la que se le agregan procedimientos para generar automáticamente el nivel de control adecuado. También, provee flexibilidad en cuanto a las máquinas y las herramientas utilizadas en la planta.

Los autores, lo clasifican como un Co-Simulador, el cual es generalmente capaz de interconectar múltiples herramientas con interfaces adecuadas, para construir un modelo virtual de la planta física. Este Co-Simulador emula la ejecución de un entorno de manufactura

real, siendo, el programa Arena-RT utilizado para la implementación del módulo de simulación, por su habilidad de sincronizar operaciones en tiempo real.

En otras palabras, el prototipo puede sincronizar la lógica de la simulación del sistema de control con procesos externos de un sistema complejo de manufactura flexible (CFMS). El sistema esta implementado mediante el acople de un controlador de simulación centralizado (CSC) y un simulador en tiempo real para cumplir con estratégicas dinámicas de control a nivel de *shop-floor*. Los autores de [56] expresan que dado que se ha encontrado que hay un número reducido de investigaciones en cuanto a la integración de simuladores modulares como meta modelos para el control de sistemas de manufactura, el enfoque de este artículo está en el desarrollo de meta modelos basados en Co-Simuladores inteligentes con reglas dinámicas para que permitan conferir flexibilidad a máquinas y herramientas. En consecuencia, el trabajo utiliza un programa de simulación para el modelado, supervisión y control de la planta física. Esta tesis y [56] comparten el uso de la sincronización y la utilización de simuladores para SCADA. A diferencia respecto a la sincronización, en esta tesis se presentan modelos genéricos que permiten implementarla con cualquier *software*, mientras que en [56] la implementación de la sincronización depende de Arena-RT, siendo en este sentido menos flexible o bien limitado al uso de este *software*. Otra diferencia es que en [56] la solución está basada en orientación a objetos (OO) (considerados normalmente entidades pasivas), mientras que esta tesis se basa en el paradigma de agentes (considerados entidades activas), que se encuentran más cercanos al dominio de la aplicación (dado el nivel de abstracción) además de proveer ciertas ventajas respecto a la OO resumiéndose en el concepto de autonomía. Ambas soluciones también difieren en cuanto a la arquitectura y a la manera de presentar los modelos. Finalmente, en esta tesis se presenta una solución que puede servir para representar entornos de realidad mixta, con ello se puede tener interacción entre simulación y realidad, característica que es muy útil al momento de estudiar diferentes escenarios con un número limitado de recursos físicos, característica no contemplada en [56].

En [57] se presenta un *framework* llamado SIMBA (Sistema Multiagente Basado en Artis) el cual se basa en un método de desarrollo llamado MESSAGE [58] al que se le agrega RT-UML [59] para crear RT-MESSAGE y una arquitectura basada en ARTIS [60] (*agent architecture for Hard real-time Systems*) para la implementación de MAS en tiempo real. RT-MESSAGE

cubre actividades de análisis, diseño e implementación de esta clase de sistemas. Por detrás de la arquitectura SIMBA se encuentra el principal componente que es el agente ARTIS, propuesto para modelar MAS en tiempo real. Finalmente, InSiDE [61], una herramienta visual para el despliegue de agentes ARTIS es utilizada para implementar esta propuesta. La novedad de la tesis presentada en [57] radica en mezclar acciones en tiempo real con técnicas de inteligencia artificial (paradigma de agentes) en un entorno distribuido. Para validar estas características el autor de la tesis usa dos demostradores, uno relacionado con la gestión de contenedores en el puerto de Valencia y el otro mediante robots móviles con un objetivo en común, mover un objeto hacia una posición final. Esta tesis es similar en cuanto a lo propuesto en [57] en relación a la implementación de MAS en tiempo real. Ambos trabajos de tesis se diferencian en cuanto a las herramientas utilizadas para su implementación, que en [57] el *framework* es validado para *hard real time* y en esta tesis para *soft real time*, pero la diferencia fundamental es que en este trabajo de tesis se le suma a la complejidad del tiempo real de MAS la característica de sincronización de simulación con realidad, teniendo de esta manera agentes con una parte virtual/simulada ejecutándose en tiempo real y sincronizada con su contraparte real.

En [62] el autor parte de una premisa y es que la información en los escenarios de negocio actuales representan el activo más valioso y estratégico para una organización. Una empresa que accede a la información correcta en el momento adecuado puede hacer que la misma se mueva a posiciones de liderazgo en el mercado. Para lograr esto, las organizaciones necesitan gestionar la información y el conocimiento derivados de ella. Los *Managers* y otros tomadores de decisiones están experimentando un efecto colateral relacionado con la enorme cantidad de información a la que tienen acceso, lo cual está afectando la toma de decisiones, lo que por el contrario, podría ser bastante similar, en cuanto al efecto, a la falta de información. La contribución de [62] pretende proveer a los tomadores de decisiones con una tecnología en la cual delegar parte de la carga del proceso de la toma de decisiones. Para la implementación, los autores utilizaron tecnología MAS para demostrar cómo la misma puede ayudar a los expertos en las ciencias de la computación a diseñar esta clase de sistemas inteligentes. El paradigma de los agentes es utilizado para permitir una descripción natural del conocimiento relacionado con la aplicación de sistemas, para permitir la realización de representaciones

complejas de conocimiento, para adaptarse elegantemente a las negociaciones a largo plazo y a los cambios continuos del entorno, los cuales, son elementos clave en el logro de las metas comprometidas en un proyecto. Para proveer un rápido prototipo de los sistemas, los autores utilizan generación automática de código orientado a agentes. Para aplicar y verificar estos objetivos, los autores usas una plataforma MAS desarrollada en JADE siguiendo la metodología GAIA en una aplicación de cuidado de la salud. Existen importantes diferencias entre este trabajo y la tesis aquí presentada, desde la aplicación final utilizada para validación, la tecnología utilizada para la implementación y el nivel dentro de la organización a la que está enfocada la solución, siendo en este trabajo de tesis más relacionado con el *shop-floor* y en [62] más bien dirigido hacia el área de *management*. Sin embargo, poseen puntos de coincidencia en cuanto a la utilización de MAS y la preocupación de proveer a las empresas acceso a la información adecuada en el momento oportuno; en esta tesis este hecho se encuentra latente en cuanto a la posibilidad de proveer pronósticos más asertivos mediante el uso de simulaciones sincronizadas con la realidad con el fin de mejorar la toma de decisiones.

En la tesis presentada en [63], el autor hace frente al problema de la agilidad a nivel de *shop-floor* utilizando un punto de vista holístico, lo cual forma parte del aspecto de novedad de este trabajo. Para ello, el autor se ve inspirado por diferentes modelos de negocio, partiendo del contexto legal usa un enfoque sistémico para ver las interconexiones entre las diferentes áreas y el *shop-floor*. Esta base es usada como una analogía para proponer CoBASA (*Coalition Based Approach for Shop Floor Agility*). Las coaliciones son básicamente acuerdos/uniones establecidas por medio de contratos individuales (en este caso entre agentes), las cuales definen que y como cada miembro está obligado dentro de la coalición. La arquitectura CoBASA usa un enfoque multiagente y extiende (agregando la fase de reingeniería) el propósito de los holones en el entorno de manufactura, cuyo foco se encuentra más bien en la fase operativa. CoBASA ha sido diseñado para soportar una evolución ágil del *shop floor* por medio de la implementación de fases de coalición y la 'agentificación' de componentes del área industrial. El trabajo propone que ante las situaciones de cambio, es más rápido realizar una reconfiguración más que una reprogramación de los componentes involucrados. CoBASA ha sido implementado en JADE y Protégé-2000 [64] como plataforma de modelado. Para realizar pruebas de factibilidad de CoBASA, una celda de producción dentro de un entorno de

manufactura ha sido desplegado, lo que en esencia representa un sistema de ensamblado (NovaFlex cell) [65]. Este trabajo (CoBASA) y la tesis aquí presentada comparten el uso de MAS aplicado al área industrial así como la creación de arquitecturas basadas en un enfoque multiagente; también ambas tesis tratan el tema de agilidad a nivel de *shop-floor* tratándolo CoBASA mediante un enfoque más holístico mientras que en esta tesis la agilidad se trata desde el punto de vista del despliegue de sistemas industriales a nivel de *shop-floor* mediante el uso de simuladores para SCADA de la planta real. Ambos trabajos utilizan diferentes tecnologías para la implementación y particularmente en la tesis aquí presentada se consideran simulaciones sincronizadas con la realidad y entornos de realidad mixta.

La tesis presentada en [66] como un compendio de publicaciones explica el gran interés de aplicar el paradigma de multiagentes al área industrial. En este trabajo, el enfoque utilizado para la investigación se basa en simulaciones ABM para la evaluación de la gestión de operaciones aplicadas a contenedores en terminales (CT) portuarias. La motivación de esta investigación es la de proveer una herramienta que ayude a *managers* de CT en dos principales intereses, servir a los barcos con contenedores lo más rápido posible mientras también se busca minimizar costes, lo cual a menudo parecen ser metas en conflicto. Para ello, el autor ha desarrollado un *software* para simulaciones llamado SimPort. SimPort ayuda a mejorar el desempeño de las CTs desde la perspectiva de los *managers* así como mejorar la comprensión de factores productivos y como ellos se relacionan uno con los otros. La implementación de los modelos se realiza utilizando un enfoque MAS, donde, cada equipo de la terminal, es representado con un agente. Los equipos pueden ser grúas, vehículos de transporte, etc. Para verificar y validar SimPort como una herramienta de modelado y simulación se implementaron diferentes modelos de terminales y se ejecutaron diferentes experimentos. Las conclusiones reportadas muestran que un enfoque basado en MAS para las simulaciones parece ofrecer a la gestión de CT una herramienta adecuada para el diseño, coordinación, control, evaluando y mejorando la productividad. Tanto en [66] como en el trabajo de esta tesis se han utilizado demostradores en el área industrial, particularmente logística (entendida rápidamente como el área el movimiento de objetos o personas de un punto a otro), en el primer caso para logística exterior y en el segundo para logística interior. Una de las diferencias principales entre ambos trabajos es que [66] está limitada únicamente a simulaciones, mientras que en esta tesis la

propuesta es que la misma simulación que ya ha sido verificada y validada pueda ser extendida para SCADA en tiempo real de la realidad, con la prestación adicional de intentar mantener la simulación sincronizada con la realidad.

A continuación se nombran brevemente algunos trabajo en el que esta tesis tiene algún punto de coincidencia, por ejemplo con INGENIAS [67], [68], el enfoque en común es en que los modelos deben tener un nivel de abstracción independiente de la plataforma de implementación. Desde el punto de vista de aplicaciones basadas en MAS para el transporte de objetos en plantas industriales, el demostrador con AGVs utilizado en esta tesis puede ser comparado con [69]–[75], donde también AGVs son utilizados para este propósito. El autor de la tesis también está de acuerdo con [56], [76], [77], en que hay una presencia limitada en cuanto al número de trabajos publicados que integran simuladores modulares como meta modelos para el control de plantas industriales, lo cual se vuelve aún más limitado si se consideran a los que están basados en MAS. Este trabajo comparte con [78], el uso de un enfoque basado en ABM para el desarrollo de entornos que utilizan sistemas basados en agentes y con [37], acerca de la idea de fusión de datos *on-line* de simulaciones con datos provenientes de la realidad.

Respecto a otros trabajos fuera del enfoque de simulación ABM, tales como [79]–[86], los mismos muestran que se pueden implementar similares soluciones, pero con las limitaciones derivadas de no utilizar el paradigma de agentes y muchas veces con un trabajo redundante dado que en la mayoría las simulaciones son utilizadas para las primeras fases de diseño y luego los algoritmos son reescritos o bien generados automáticamente para el despliegue final.

Al mismo tiempo que la tecnología basada en agentes, comienza a ser adoptada, existen algunas tendencias a nivel industrial como las llamadas fábricas virtuales [87], entornos tipo *Mixed Reality Agent* (MiRA) [88], y las también llamadas ‘industria 4.0’ [89], las cuales todavía están en sus comienzos.

Tal como se ha mostrado a lo largo de este estado del arte, afortunadamente existen varios trabajos que promueven el uso del paradigma MAS como la mejor alternativa para hacer frente a problemas que se presentan en sistemas complejos, como por ejemplo, la flexibilidad, la adaptabilidad, la robustez, etc. Desafortunadamente, no existen todavía suficientes

demostradores que justifiquen una amplia adopción de esta tecnología a nivel industrial y en otras clases de empresas. Este es un problema común en la comunidad investigadora y científica; cómo acelerar la adopción de un 'nuevo' paradigma tecnológico en el mundo de las empresas, hecho que lleva siempre su tiempo, aún cuando la tecnología es suficientemente madura y cuyas ventajas para hacer frente a la complejidad de los sistemas está demostrada. Este problema puede ser encarado con la adopción de técnicas y estándares que disminuyan el riesgo inherente a la adopción de una 'nueva' tecnología y, quizás, introducir la misma como una extensión incremental de métodos conocidos y ya testeados.

Tal como se ha venido mostrando, los sistemas actuales son complejos, y en el futuro, las tendencias indican que continuarán incrementando su complejidad. Esto podría estar sucediendo, debido a las necesidades de los consumidores, de adquirir productos personalizados. Los mismos podrían crear sus propios productos desde cero, o quizás, partir de un diseño que ha sido compartido en la nube, luego de esto, las órdenes de producción podrían ser enviadas a la nube, donde, alguna empresa suficientemente competitiva, significando con ello, con suficiente flexibilidad, adaptabilidad y un coste competitivo de sus sistemas de producción decidirá producir la orden de este cliente. Una vez las órdenes han sido producidas, tienen que ser enviadas a los clientes en tiempo y nuevamente a un coste competitivo, lo cual nuevamente incrementa la complejidad de la logística.

Además, la llamada industria 4.0 [89]–[92], la cual cubre el realineamiento de las investigaciones en campos como la producción, los servicios y el diseño del trabajo, también se enfoca en temas tales como '*Smart Factories*', lo que significa, sistemas inteligentes para producción y procesos, así como la realización de redes distribuidas y centros de producción en red. Al mismo tiempo, esta aborda el campo del Internet de las cosas (IoT) y el llamado Internet de las cosas a nivel industrial (I2oT), mientras que '*Smart Production*' pone el foco en áreas tales como el uso de 3D en aplicaciones industriales, la logística interna a nivel de producción y la interacción hombre-máquina.

Estas tendencias se suman al escenario actual industrial, protagonizado por productos a nivel masivo 'económicos' y productos personalizados 'caros', los cuáles se ven influenciados por

tendencias como las de ‘pequeñas fábricas en el hogar’, basada en el concepto de ‘hazlo tu mismo’.

Para poder abordar la complejidad de lo expuesto en los párrafos anteriores, es conveniente utilizar paradigmas con niveles de abstracción superiores, tales como el de los agentes y el de los holones ya que por su abstracción se encuentran más cercanos al dominio de la aplicación, en vez de utilizar paradigmas que ya han sido aceptados ampliamente, como el de la orientación a objetos.

Por otro lado, la capacidad de cómputo de los ordenadores se ha ido incrementando a lo largo de la última década así como su coste decreciendo. Este hecho podría habilitar el uso de programas altamente demandantes de capacidad de cómputo, como por ejemplo los simuladores, los cuales, desde sus comienzos estuvieron limitados (debido a su gran coste computacional) para las primeras etapas involucradas en el desarrollo de un sistema complejo.

Finalmente, en este capítulo se ha realizado una revisión del estado del arte, se han encontrado y propuesto tendencias a nivel industrial y se ha comparado el trabajo de esta tesis con otras publicaciones científicas y tesis doctorales.

El próximo capítulo se enfoca en proveer el marco conceptual que da soporte a esta tesis.

3 Marco conceptual

En este capítulo se presentan de manera sintética dos principales conceptos que se encuentran directamente relacionados con el trabajo expuesto en esta tesis. Para ello, se los ha dividido en dos grupos, el primero relacionado con paradigmas de programación y el segundo donde se presenta información relativa a dos tipos de programas para ordenadores diferentes, siendo estos, los simuladores y los *frameworks*. En esta última sección, se pone un mayor énfasis en los simuladores, por ser la tecnología *software* utiliza en este trabajo.

3.1 Agentes, objetos y holones

En esta sección se pretende proveer al lector de una visión sintética de los aspectos más relevantes de tres paradigmas de programación, como lo son el de los objetos, agentes y holones. Principalmente se describirá el enfoque orientado a agentes, que es el utilizado en esta tesis, y de manera aún más concisa se nombrarán los otros dos paradigmas.

En la literatura existen diversas definiciones de los agentes [22], [93], sin embargo, en general, se podría decir que un enfoque simple permitiría asociarlos como entidades con un cierto grado de inteligencia, autonomía, proactivas y que interactúan con el entorno a través de sensores y actuadores.

El paradigma basado en agentes puede ser contemplado desde dos puntos de vista. El primero asociado a las características individuales de estas entidades (enfoque microscópico) y el segundo, en relación al comportamiento de un sistema integrado por múltiples agentes (enfoque macroscópico).

Desde el punto de vista microscópico, los agentes presentan características individuales, que han sido expuestas en [94] y que intentan responder a la pregunta de qué es un agente en función de las propiedades conferidas a estas entidades por diferentes investigadores del área, las cuales se sintetizan a continuación.

La principal característica distintiva de un agente es su autonomía, lo que significa que puede operar sin la intervención directa de otros agentes o humanos y que tiene algún tipo de control

sobre sus acciones y su estado interno; la reactividad es entendida como la capacidad de percibir su entorno y responder a los cambios que acontecen en un contexto temporal adecuado; la proactividad implica que no solo reacciona al entorno, sino que es capaz de mostrar un comportamiento basado en metas y es capaz de tomar la iniciativa; la habilidad social implica la interacción con otros agentes por medio de algún tipo de lenguaje de comunicación de agentes (ACL). Es común también dentro del área de inteligencia artificial (AI) conferir a los agentes características relacionadas con los humanos tales como nociones mentales (conocimientos, deseos, intenciones, creencias, obligaciones, etc.), yendo algunos más lejos considerando en los agentes estados emocionales. Existen otros atributos que han sido asociados a los agentes tales como movilidad, siendo la capacidad del agente de moverse a través de una red electrónica; veracidad, en la que se asume que el agente no comunicará una información falsa a conciencia; benevolencia, asociado a la carencia de metas conflictivas y que el agente hará siempre lo que se le ha solicitado; finalmente, la racionalidad se entiende como la capacidad del agente de actuar para lograr sus metas y que el mismo no actuara (mientras sus creencias se lo permitan) en contra o para evitar que las metas sean logradas.

En el párrafo anterior se han descripto varias características individuales asociadas con la noción de agentes, más bien relacionadas con aspectos teóricos de cómo se debería razonar respecto a estos, constituyendo de este modo una especie de especificaciones. Para llevar este razonamiento a la implementación, hacen falta dos elementos, el primero referido a la arquitectura lo que implica la manera de cómo organizar (división en módulos) y construir (cómo los módulos deberían interactuar) los agentes, y el segundo, en relación a los lenguajes de programación orientados a agentes, que deberían permitir implementar las teorías o razonamientos asociados a estos, al igual que la arquitectura deseada.

En cuanto a las arquitecturas, según [94], se pueden dividir en deliberativas, reactivas e híbridas.

Un agente con arquitectura deliberativa es aquel que tiene representado de manera explícita un modelo simbólico del mundo y donde las decisiones son realizadas por medio de un razonamiento simbólico. En este sentido, los agentes se comportan acorde a cómo piensan, chequeando a través de un espacio de posibles comportamientos, manteniendo un estado

interno y prediciendo los efectos de las acciones. Dentro de este tipo de arquitectura, se encuentran dos que son las más conocidas; la primera asociada a un agente que es capaz de realizar una planificación adecuada según la meta especificada y la segunda, asociada a agentes con un razonamiento asociado a un modelo de creencias, deseos e intenciones, del inglés BDI. Este tipo de arquitectura fue la inicialmente usada en la comunidad de la inteligencia artificial, dado que en la década de 1980 los ordenadores estaban limitados a un procesador, siendo esta restricción una plataforma natural para este tipo de agentes que utilizan un modelo de cómputo secuencial. En consecuencia, la suma de lo citado anteriormente conducía a obtener agentes muy demandantes de recursos computacionales, los cuales se volvían inadecuados cuando las aplicaciones poseían restricciones temporales.

La arquitectura reactiva embebida en un agente ayuda a que este pueda reaccionar de una manera más rápida, convirtiéndose en una opción válida para aplicaciones de tiempo real. Este tipo de enfoque se basa más bien en que el agente 'descubre' el mundo a través de los sensores y en consecuencia va reaccionado y modificando el entorno a través de actuadores, a diferencia de la arquitectura deliberativa, no necesita de una representación explícita del mundo ni tampoco de un razonamiento simbólico, convirtiéndola en una opción ideal cuando el entorno es desconocido (por lo tanto no puede ser representado) y tiene que ser descubierto. Dado que las decisiones no son tomadas por modelos complejos de razonamiento, éstas pueden ser atendidas con mayor rapidez y en este sentido posibilitar su ejecución en un tiempo más acotado.

La arquitectura híbrida está integrada por una arquitectura deliberativa y otra reactiva. Surge como una solución que permite explorar un balance entre ambos extremos y donde el diseñador según las especificaciones de la aplicación podrá elegir qué procesos se resolverán en cada capa. Su aplicación/implementación ha sido posible gracias a los avances tecnológicos en cuanto a capacidad de cómputo de los ordenadores y demás dispositivos, ya que en este sentido se deben computar (incluso de manera paralela) ambas capas. Esta organización permite que el agente reaccione rápidamente a eventos del entorno y además pueda gestionar información en un nivel superior de abstracción para cumplir con metas a más largo plazo, llevada a cabo por la capa superior o deliberativa.

Definida la arquitectura, llega el momento de implementarla por medio de algún lenguaje de programación orientado a agentes. En este sentido, la solución puede ser llevada a cabo utilizando *frameworks* orientados a agentes o bien programas de simulación basados en agentes; en ambos casos la utilización de uno u otro será decisión de las personas encargadas del diseño y el despliegue y la solución adoptada variará acorde a la aplicación y sus especificaciones. Existe una gran variedad de programas que se pueden utilizar para implementar la solución, alguno de los cuales aparecen listados en [95]–[97].

Hasta el momento se ha tenido en cuenta las características individuales de los agentes, sin embargo, existen otros dos paradigmas cercanos, comparables e incluso a veces que pueden crear cierta controversia/confusión entre los ingenieros en programación, siendo estos la orientación a objetos (OO) y a holones (HO).

Básicamente, ya que el paradigma de OO es todavía el más ampliamente utilizado, se crea una cierta confusión entre los que son los objetos y los agentes. Esto viene ocasionado por el hecho que ambos representan una encapsulación de algún concepto abstracto. Si embargo, en [63] se cita que la orientación a agentes puede ser considerada como una extensión de la orientación a objetos, donde un agente es diferente a los objetos en los siguientes aspectos:

1. Alto grado de autonomía. Mientras que los objetos no pueden tener control respecto a cuándo y cuáles objetos llaman a sus métodos públicos, un agente tiene un completo control sobre ello. Los agentes deciden por si mismos cuando pueden llevar a cabo sus acciones.
2. Alto grado de proactividad. Mientras que algunos objetos exhiben iniciativa sobre algunas metas, la mayoría de estos son más bien pasivos que activos. Los agentes, por el contrario, siempre muestran algún tipo de comportamiento deliberativo.
3. Actividad social. Los objetos no han sido pensados para mostrar una actividad social tal como sucede con los agentes. La cooperación que exhiben los objetos está limitada a mensajes que se envían entre ellos. Claro está, que es posible construir complejas interacciones de intercambio de mensajes entre estos, sin embargo, esto no está en las bases de los conceptos de los objetos.

4. Hilo de ejecución controlado. Mientras que algunos objetos del sistema exhiben esta propiedad (objetos activos) y, en este sentido tienen algún grado de autonomía, esta no es la situación más común. Sin embargo, cada agente posee su propio hilo de ejecución/control.

El otro paradigma citado que presenta ciertas similitudes con el de agentes y por ende emergen algunas discusiones sobre el tema es el de orientación a holones. Este enfoque fue introducido por Arthur Koestler en 1967 [98]. La palabra *holon* es una combinación de la palabra griega *holos*, que significa todo y el sufijo *on* que significa parte o partícula. Según esta visión, un *holon* es una estructura que se asemeja a sí misma y que está compuesta por por diferentes holones como sub estructuras. La jerarquía así formada y que está compuesta por holones recibe el nombre de holarquía. Esta holarquía debe cumplir tres condiciones: 1) ser estable; 2) tener capacidad de autónoma y 3) tener capacidad de cooperación.

Cuando de analiza una estructura compleja, es fácil notar que está compuesta por partes y si se adentra en un mayor detalle, se puede observar que estas partes según el punto de vista del observador representan un todo al mismo tiempo. Teniendo en cuenta la naturaleza híbrida, la capacidad de autonomía y la de cooperación, Van Brussel define en [99] un holon desde el punto de vista industrial tal como se expresa a continuación:

"*A holon is an **autonomous** and **cooperative** building block of a manufacturing system for transforming, transporting, storing and/or validating information and physical objects. The holon consists of an information **processing part** and **often a physical** processing part. A holon can be part of another holon*"

Dadas las similitudes entre los holones y los agentes, tal como se expresan también en [94], la arquitectura ALOHA presentada en esta tesis puede ser considerada tanto como un agente con una parte física o bien de acuerdo a la definición arriba citada como un holon. Durante este trabajo la mayor parte de las veces, el enfoque adoptado es el de agentes.

Hasta el momento se han tratado diferentes entidades contemplando sus aspectos básicamente como individuos, principalmente a los agentes. Si bien por si solas son suficientes para resolver algunas tareas, su potencial se ve incrementado exponencialmente cuando los agentes actúan en conjunto, permitiendo de esta manera que emerjan situaciones asociadas a las

interrelaciones entre estos. Para que los agentes puedan interactuar tienen que establecer algún tipo de comunicación entre ellos, es aquí cuando cobra importancia el lenguaje de comunicación de agentes, del inglés ACL. Para que este ACL no sea visto como un simple protocolo de intercambio de mensajes, la semántica del discurso se basa en la teoría de los actos del habla (*speech act theory*). Dentro de los ACL más conocidos se encuentran FIPA-ACL, KQML y KIF.

Con todo, se han mostrado las características relevantes e individuales de los agentes y se han nombrado los ACL relacionados con los actos comunicacionales entre agentes, los cuales les permiten compartir e intercambiar mensajes/conocimientos cuando estos se encuentran en un entorno multiagente. De esta manera, al conjunto/sociedad de agentes se los denomina un sistema multiagentes (MAS).

En los MAS, los agentes coordinan su conocimiento, las actividades y razonan sobre el proceso de coordinación. Los agentes pueden ser entidades físicas o virtuales con capacidad de actuar, percibir el entorno y comunicarse con otros agentes. Un MAS está compuesto por una red de agentes los cuales están acoplados de manera flexible como una única entidad similar a una sociedad, para resolver un problema que no puede ser resuelto por un único agente. Una de las lecciones más importantes a nivel general sobre MAS radica en que es de crítica importancia comprender el **tipo** de interacción que tiene lugar entre los agentes [22]. Un tipo de relación que se puede dar en MAS es el de coordinación, que a su vez se puede dividir en diferentes técnicas [100], donde se presenta como hipótesis que todo mecanismo de coordinación puede ser expresado a través de cuatro estructuras principales, los compromisos, los acuerdos, las convenciones sociales y las capacidades de razonamiento locales. En [63], la coordinación en MAS se divide en dos grupos principales, el primero asociado a la cooperación y el segundo a la competencia, ambos ramificándose en otros sub grupos.

El enfoque MAS es inherente a la inteligencia artificial distribuida (DAI) (según clasificación en ACM), donde se pretende aprender a resolver problemas complejos, hacer frente a cuestiones de planificación e investigar problemas relacionados con la toma de decisiones.

Con todo, el enfoque a nivel agente (microscópico) se puede comparar a las características individuales de las personas, mientras que un MAS (macroscópico), está más relacionado con una visión de sociedad en donde viven los humanos.

3.2 Simuladores y *frameworks*

En esta sección se pretende nombrar inicialmente algunas diferencias entre los programas denominados *frameworks* y aquellos utilizados para simulación. Dado que esta tesis está basada en simulación, luego de un pequeño estado del arte sobre estos programas, se sintetiza un marco conceptual genérico relacionado con estos programas.

En el estado del arte, se ha observado que existe una tendencia a confundir o a utilizar de manera análoga a los programas de simulación y a los programas denominados *frameworks*. Un *framework* es básicamente un entorno de trabajo, pudiendo ser este de propósito general o bien específico. Los *frameworks* no están pensados para proveer por defecto las características de la simulación, como por ejemplo la animación que representa la ejecución de un modelo, un entorno que permita acelerar o ralentizar el tiempo, etc.; es por ello, que los *frameworks* son provisto de las características de la simulación a través de *plugins*, que confieren esta característica extendida. Normalmente, en estos programas se construye el código utilizando lenguajes tales como Java, C o C++, Python, etc.; lo que muchas veces limita su utilización a ingenieros con conocimientos de programación. Dada la naturaleza para la que han sido pensados estos programas, la utilización de *frameworks* para implementar soluciones es menos demandante de recursos de cómputos que los simuladores. Resulta lógico o más bien aventurado para un programador pensar en construir un simulador *ad-hoc* utilizando como herramienta algún *framework* que en sentido inverso, utilizar un programa de simulación para crear un programa que puede ser considerado un *framework*. En el estado del arte existen una gran cantidad de *frameworks*, los cuales están basados en diferentes paradigmas de programación Qt (Objetos), (JADE, JACK, Mobile-C) (Agentes), JANUS (Holones), por nombrar solo algunos que son de propósito general. En cuanto a aquellos *frameworks* que son de propósito más específico, se pueden nombrar algunos relacionados al área de robótica, como lo son ROS, YARP, Open-RTM, Open-RDK, OROCOS y ORCA, los cuales ofrecen

un *middleware* para robótica distribuida, son de código abierto y de libre uso. En todos los casos, excepto para OROCOS, se proveen extensiones para conferir la capacidad de simulación, las cuales están principalmente provistas por programas de simulación tales como Stage o Gazebo. En ninguno de los casos, estos *frameworks* de robótica permiten una programación basada en agentes, por lo tanto para ser utilizados con este paradigma, se les debería agregar esta característica, tal cual se hace con la simulación. Un estudio más detallado sobre *frameworks* específicos para robótica se puede encontrar en [101].

Por otro lado, los programas de simulación están pensados para imitar la ejecución de un proceso o sistema del mundo real a través del tiempo [102]. Para poder simular antes se necesita implementar un modelo que representa las principales características físicas o bien abstractas del sistema o los procesos. Por lo tanto, el modelo representa al sistema por sí mismo, mientras que la simulación es la ejecución de estos modelos utilizadas para mostrar la operación del sistema en función del tiempo [103]. Normalmente, las simulaciones se utilizan para estudiar diferentes escenarios y en consecuencia poder mejorar la toma de decisiones. Poseen una series de herramientas que posibilitan ver fácilmente la ejecución de los modelos a través de animaciones, así como también tener el control del tiempo de la simulación (acelerar, ralentizar, elegir entre tiempo discreto (*ticks*) o continuo), etc. Al igual que en los *frameworks*, los simuladores pueden ser de propósito general o bien de propósito específico y estar basados en diferentes paradigmas. Un tipo especial de programa de simulación es aquel que está basado en el paradigma de agentes, permitiendo de esta manera realizar modelado basado en agentes (ABM) y, en este sentido, existen numerosos programas como los listados en [96], pudiendo considerarse del tipo genéricos a MASON, Swarm, Netlogo, AnyLogic, Repast, etc., los cuales han sido comparados en [104]; mientras que de tipo específico se pueden nombrar a TerraME, MobiDyc, MATSim, etc. Un listado más completo y con comparativas entre plataformas tanto de *frameworks* como simuladores se puede encontrar en [95], [97], [105], [106]. Cabe destacar que hasta el momento todos los simuladores nombrados están basados en ABM; con el objetivo de proveer un estado del arte más amplio sobre esta clase de programas cabe citar algunos no basados en ABM, tales como Arena, AutoMod, Flexim, Witness, Webots, V-REP, Gazebo, Stage/Player, etc., siendo los 5 primeros programas de pago.

Dado que la simulación es la tecnología que se utiliza en esta tesis, a continuación se incluyen de manera sintética algunos conceptos generales relacionados con estos programas, los cuales se basan en las siguientes referencias [102], [107]–[112], y las también citadas en estos trabajos.

Antes de profundizar en conceptos como modelos, simulación, análisis, etc., resulta conveniente definir lo que es un sistema, ya que este concepto puede significar diferentes cosas según lo que marca cada disciplina. Una definición consensuada respecto a lo que es un sistema ha sido desarrollada por INCOSE. Este organismo sugiere que un sistema es una construcción o colección de diferentes elementos que juntos producen un resultado que no puede ser obtenido por uno de los elementos por si solo. Los elementos pueden incluir personas, hardware, software, instalaciones, etc.; en resumen todo tipo de cosas requeridas para generar cualidades, propiedades, funciones, características, comportamientos y un desempeño a nivel de sistema. Es importante destacar, que el valor del sistema como un todo radica en las relaciones entre sus partes. Los sistemas pueden ser de tipo físicos, algo que ya existe, o bien hipotéticos como un plan o un concepto de algo físico que no existe. Por otro lado, los sistemas pueden ser de naturaleza discreta, continua o mixta. En los discretos, las variables de estado, las cuales describen completamente al sistema en cualquier momento del tiempo, cambian instantáneamente en puntos separados/no continuos en el tiempo; por ejemplo el número de personas que entran y salen de un banco. En los sistemas continuos, las variables de estado cambian constantemente en función del tiempo; por ejemplo el volumen del agua fluyendo en un rio cuando llueve. En el caso mixto, existen variables del sistema que cambian de manera discreta y otras continuas, por ejemplo, la variación en función del tiempo de los vehículos disponibles en un sistema de transporte (discreto) y la velocidad/distancia de los mismos durante el recorrido en función del tiempo (continuo).

Teniendo en cuenta la naturaleza de los sistemas y sus características, se pueden construir aproximaciones del mundo real, denominadas modelos. Conceptualmente, un modelo es una representación física, matemática o de alguna manera con cierta lógica, de un sistema, entidad, fenómeno o proceso. Simplemente, los modelos sirven para representar eventos o cosas que son reales o artificiales. Existen diversas razones por las cuales se realizan modelos de sistemas existentes, entre ellas, debido a que resulta difícil o imposible de investigarlos por diferentes

razones. Para crear el modelo, el diseñador mediante un proceso de abstracción extrae de la realidad una descripción del sistema. Sin embargo, es importante destacar que modelar no significa representar todos los aspectos del sistema a ser estudiado. Esto podría ser muy demandante de tiempo, caro y complejo. Por ende, el modelo debería desarrollarse de la manera más simple posible, representando solo aquellos aspectos que se desean investigar. En consecuencia, el modelo puede representar al sistema desde algún punto de abstracción o bien a través de múltiples niveles de abstracción, con la meta de obtener una representación del sistema que sea de alguna manera confiable. Decidir que aspectos del sistema necesitan ser incluidos en el modelo constituye uno de los desafíos del diseñador/modelador. Al construir el modelo, este puede ser de tipo físico, por ejemplo un modelo a escala de un coche para estudiar su aerodinámica, o bien del tipo hipotético, siendo este un conjunto de ecuaciones matemáticas o sentencias lógicas que describen el comportamiento del sistema.

Al momento de implementar los modelos en la simulación, hay que considerar dos aspectos diferentes, uno es aquel relacionado con el entorno/sistema modelado y simulado y el otro respecto a la ejecución del modelo del entorno de simulación [109]. El primer caso se refiere a los diferentes tipos de modelos tales como los listados en [107], entre los que también se encuentran los ABM. El segundo caso, se refiere a los modelos de cómputos (MoC) soportados por el entorno de simulación, donde en aquellas simulaciones consideradas híbridas se integran diferentes MoC con diferentes tipos de modelos. En la Fig. 1.6 de [111] se muestra un listado incompleto, pero útil en cuanto a la diversidad de MoC disponibles. Por lo tanto, para ejecutar un modelo, los entornos de simulación requieren que el modelo sea descripto en un formalismo de simulación en particular. Por ejemplo, los simuladores híbridos soportan modelos definidos en términos de variables de estado, ecuaciones, eventos en el tiempo, eventos de estado, etc. Es por ello, que un modelo que se describe en términos de los conceptos de un dominio específico, debe ser traducido en los conceptos de un formalismo de simulación antes de que pueda ser ejecutado en un entorno de simulación específico [109].

Dentro de los diferentes tipos de modelos, es de especial interés aquel denominado ABM, dado que es el utilizado durante este trabajo. Los ABMs representan un importante paradigma de modelado cuyo origen se encuentra en la investigación de diferentes tipos de fenómenos tanto humanos como sociales [107]. La idea importante aquí es que un ordenador sea capaz de crear

un sistema complejo por sí solo, siguiendo una serie de reglas o directivas sin tener definido este sistema de antemano por un humano. Los ABMs están compuestos por agentes, definidos en este sentido como entidades autónomas software que interactúan con el entorno y con otros agentes para lograr alguna meta o completar alguna tarea. La autonomía es probablemente el concepto más importante en estas entidades. Así, los agentes actúan en su propio interés independientemente del control de otros agentes del sistema, lo cual, no quiere decir que no sean influenciados por estos. Dada su autonomía, cada agente decide por sí mismo qué, cuándo y cómo llevará a cabo su cometido. Estas decisiones que se basan en el comportamiento del agente, están incorporadas dentro de este por su diseñador. El entorno de los agentes y la existencia de otros agentes en el mismo también juega un rol clave en la manera en la que un agente podría comportarse. Un agente está embebido con la habilidad de descubrir el entorno mediante sensores y responder a los cambios en este, o bien, simplemente observar dichos cambios esperando que uno específico acontezca. También cada agente tiene consciencia sobre los otros agentes, los cuales podrían ser monitoreados para saber qué están haciendo y posiblemente comunicarse con ellos para solicitarles que lleven a cabo alguna tarea. Lo detallado tiene una cierta analogía a la manera en la que las personas interactúan con lo que les rodea y en relación a las otras personas.

En el contexto del modelado, los ABMs pueden ser usados como un medio de experimentación para ejecutar y observar simulaciones basadas en agentes. Estos, pueden ser pensados como un laboratorio en miniatura, donde los atributos y comportamientos de los agentes, y el entorno en el que están inmersos, puede ser alterado y las repercusiones observadas por medio de múltiples ejecuciones de la simulación. La habilidad de simular acciones individuales de múltiple diversidad de agentes y medir el comportamiento del sistema resultante y sus resultados en función del tiempo, significa que los ABMs pueden ser una herramienta útil para el estudio de sistemas complejos, tales como los industriales, geoespaciales, sociales y los asociados a las ciencias naturales, etc.

Entre las ventajas de los ABMs se pueden citar tres características que los hacen especialmente adecuados respecto a otros paradigmas o MoC, siendo: 1) permite capturar fenómenos emergentes; 2) provee un entorno natural para el estudio de determinados sistemas; y, 3) es

una herramienta flexible particularmente en relación al desarrollo de sistemas con cierta complejidad.

La emergencia es un fenómeno, junto con otros comportamientos que resultan inesperados o poco familiares a las ciencias clásicas, tales como la auto-organización, el caos, la capacidad de adaptación, etc. los cuales son característicos de los sistemas complejos. Los fenómenos emergentes están caracterizados por patrones macroscópicos estables que nacen de las interacciones locales de entidades individuales. Por definición, los fenómenos emergentes no pueden ser reducidos a las partes del sistema; dado que el todo es mayor a la suma de las partes. Por lo tanto, este fenómeno puede exhibir propiedades que están disociadas de las propiedades de las partes del sistema. Estas características hacen que los fenómenos emergentes sean difíciles de comprender y predecir, particularmente aquellos resultados emergentes que pueden ser contra intuitivos.

En muchos casos, ABM es un método natural para describir y simular sistemas de ciertas características, como aquellos compuestos de entidades de la realidad. El enfoque basado en agentes es más afín a la realidad que otros paradigmas de modelado, siendo inherentemente más adecuado para la simulación de personas y objetos de una manera muy realística. En particular, el enfoque basado en agentes puede ser más útil cuando describe de manera más natural las entidades que constituyen un sistema bajo alguna de las siguientes condiciones: 1) el comportamiento individual no puede ser claramente definido, por ejemplo una multitud en pánico y huyendo; 2) ante la complejidad de un comportamiento individual. Aunque cualquier proceso puede ser explicado con una ecuaciones, la complejidad de las ecuaciones diferencias aumenta exponencialmente a medida que la complejidad de un comportamiento se incrementa. Por ende, describir el comportamiento individual por medio de ecuaciones puede convertirse en inviable; 3) las actividades son la manera más natural de describir un sistema constituido de procesos; y, 4) El comportamiento del agente es estocástico. Dado que puntos aleatorios pueden ser aplicados estratégicamente en el modelo del agente.

Finalmente, la flexibilidad en los ABMs puede ser vista desde diferentes ángulos, como pueden ser: 1) los agentes tienen la habilidad de moverse físicamente dentro de un entorno en diferentes direcciones y velocidades, lo que lo hace muy flexible en términos de variables y

parámetros que pueden especificarse; 2) los agentes se pueden especificar como una vecindad o grupo usando diversos mecanismos; 3) los ABMs pueden regular su comportamiento en función de diversos tipos de interacciones; 4) el ABM provee un marco de trabajo robusto y flexible para sintonizar la complejidad de los agentes y 5) finalmente otra dimensión que provee flexibilidad es la habilidad de ajustar diferentes niveles de descripción y agregado, pudiendo estos coexistir dentro del modelo bajo diferentes combinaciones. En consecuencia, el ABM se puede utilizar cuando no se conoce el nivel adecuado de descripción o complejidad, y la búsqueda de un nivel adecuado se obtiene por exploración. Dicha exploración se realiza ejecutando bajo diferentes escenarios los modelos, lo que es conocido como simulación.

La simulación, es el proceso de ejecutar un modelo de un sistema real y llevar a cabo experimentos con este modelo con el propósito de entender el comportamiento del sistema y evaluar varias estrategias de operación del mismo. La simulación se ha convertido en una técnica de estudio ampliamente utilizada en estos días dada su gran prestación para resolver problemas de sistemas complejos que no pueden ser analizados usando otras técnicas. Es por ello que a continuación se citan principalmente las ventajas y algunas de las desventajas. Entre las ventajas de la simulación se pueden citar: 1) se pueden explorar nuevas políticas, procedimiento operativos, reglas de decisión, estructuras organizacionales, flujos de información, etc. sin interrumpir las operaciones en curso; 2) las hipótesis acerca de cómo y por qué ciertos fenómenos ocurren pueden ser testeados para viabilidad; 3) se pueden identificar cuellos de botella en el flujo de materiales, información, y productos; 4) puede ser una prueba invaluable para comprender cómo el sistema opera realmente en contraste a cómo todos creen que éste funciona; 5) se pueden testear nuevos diseños, organizaciones físicas, programas de ordenadores, sistemas de transporte, etc. antes de asignar recursos a su adquisición y/o implementación. Sin embargo, existen algunas desventajas de la simulación, como: 1) la construcción de modelos requiere de un entrenamiento especializado. La calidad del análisis depende de la calidad del modelo y la experiencia/destreza del modelador; 2) los resultados de la simulación pueden ser a veces difíciles de interpretar y 3) el análisis puede ser demandante de tiempo y caro.

Por otra parte, uno de los aspectos importantes de la simulación, es aquel relacionado con la variable independiente que hace avanzar la ejecución del modelo; con esto se pretende hacer

referencia a la noción de tiempo. El tiempo, es importante de acuerdo al tipo de modelo de cómputo, siendo algunos de éstos independientes del tiempo tales como los de flujos de datos y las máquinas de estados finitos, mientras que otros dependen fuertemente del tiempo, como lo son los modelos de eventos discretos y los modelos de tiempo continuo. Tanto los modelos dependientes o independientes del tiempo pueden ser entrelazados en jerarquías, donde hay algunas combinaciones que pueden no tener sentido y otras que resultan especialmente útiles. El tiempo puede avanzar tanto de manera uniforme como no uniforme, puede ser suspendido en todo el modelo o en un sub modelo. El tiempo, también puede avanzar a un ritmo diferente en los niveles de una jerarquía. Dicha característica, es particularmente útil para el modelado de sistemas distribuidos, donde, mantener una base de tiempo perfectamente coherente no es físicamente posible, esto es conocido como '*multiform time*', y permite un modelado altamente realista que reconoce explícitamente que el tiempo no puede ser medido de manera perfecta. En los sistemas con temporizado lógico, el tiempo avanza como una secuencia de pasos discretos, llamados reacciones o *ticks*. Dado que los pasos son ordenados, no existe necesariamente una noción de 'retardo del tiempo' entre los pasos, tal como sucede en los sistemas de tiempo discreto; y tampoco hay una noción *a priori* de tiempo real. Por lo tanto este tiempo ya definido es conocido como *logical time*. *Execution time*, hace referencia al tiempo que ha tardado en ejecutarse el modelo durante la simulación. *Model time*, representa el tiempo simulado que ha transcurrido dentro del modelo, por ejemplo, la simulación de dos horas en el tiempo del modelo puede estar representada por 5 segundos de *real time*. El tiempo real, se define como aquel tiempo externo al modelo y que puede ser medido con un reloj. En simuladores de tiempo real, se requiere que el tiempo de ejecución sea lo suficientemente rápido, como para que el tiempo del modelo, pueda acercarse al tiempo que transcurre en el sistema real, si se pretende que ambos se ejecuten concurrentemente. De esta manera, teniendo ya descriptas las diferentes nociones de tiempo en la simulación, se puede comenzar el proceso de conocido como análisis.

El análisis es el proceso que se utiliza para obtener información sobre lo que se está modelando. Cuando se analiza un modelo, se considera la robustez y completitud del mismo así como el comportamiento de este al ejecutarse dinámicamente. Estando ya concebido el modelo, la primera forma de análisis es la verificación, que busca responder a la pregunta ¿Ha

sido el modelo construido correctamente?, ¿El código de simulación implementa correctamente el modelo?. Es por ello, que durante este proceso se procede a asegurarse que los requerimientos del modelo conceptual han sido adecuadamente traducidos en un modelo del sistema que puede ser ejecutado y que el modelo es capaz de ejecutarse sin errores imprevistos y con resultados confiables. La verificación se puede realizar de tres manera diferentes, con herramientas de depuración automatizadas, revisión manual del código y *tests* de estrés del modelo. En el primer caso, las herramientas de depurado proveen el soporte para encontrar y reparar errores de sintaxis y de lógica, pero resulta difícil encontrar errores semánticos. El segundo caso consiste en que la revisión del código es realizada con pares a los que se les explica el código quienes pueden realizar críticas y encontrar problemas. Esto puede ser útil sin embargo consume mucho tiempo y se requiere que todos mantengan la atención durante todo el proceso. Los *tests* de estrés del modelo son llevados a cabo ejecutando el modelo con un conjunto de entradas cercanas a las condiciones límite de máximos y mínimos con el objetivo de observar que su comportamiento es según lo esperado.

La validación, busca responder preguntas tales como ¿Se ha construido el modelo correcto?, ¿Está codificado lo correcto?, ¿Cuán bien el modelo coincide con la realidad en el contexto del propósito del modelo?. Responder a estas preguntas, principalmente consiste en asegurarnos que el modelo es robusto, que provee resultados útiles y que es una aproximación suficientemente cercana al sistema objetivo (el que se está modelando). Este tipo de análisis es realizado llevando a cabo *tests* estadísticos sobre la diferencia entre la salida del modelo y la salida del sistema objetivo. Si el sistema objetivo no existe, se podría utilizar un sistema similar para contrastar el modelo, el conocimiento de un experto en el sistema o bien un sistema podría examinar las salidas del modelo y proveer información sobre la validez del modelo, en ambos casos utilizando las experiencias pasadas como base para la toma de decisiones. Este tipo de validación subjetiva es llamada '*face validation*'. Algunos autores incluyen en este análisis un tercer elemento que consiste en el proceso de acreditación que consiste en una certificación oficial por una autoridad competente de que el modelo es aceptado para un uso específico. La autoridad puede ser un organismo o una persona responsable en cuanto a los resultados relacionados con la utilización del modelo. Por ende, esta autoridad debe ser independiente del desarrollador del modelo o simulación. La

acreditación no es una aprobación de propósito general, dado que cada modelo se acredita para un propósito o uso específico. Una representación de algunas técnicas de V&V se pueden encontrar en la Tabla 1.4 de [107].

Una vez verificado y validado el modelo, se está en condiciones de realizar las simulaciones. Cuando se simula el modelo bajo diferentes condiciones de entradas se dice que se están llevando a cabo experimentos con el modelo. Para reducir el tiempo y esfuerzo asociado a la simulación de los experimentos, es conveniente diseñarlos previamente, definiendo la información que se necesita reunir de cada simulación, el número de réplicas que se necesitan realizar, que parámetros del modelo se deberían cambiar y ser comparadas. En el caso que el modelo contenga procesos estocásticos y que se deban considerar variaciones en muchos parámetros, se podrían necesitar millones de réplicas de la simulación. Una réplica es una ejecución del modelo desde el inicio hasta el fin en un tiempo generalmente más rápido al tiempo real. Si el modelo es muy complejo, podría llevar varias horas o días entre ejecuciones. Es por ello que un diseño estratégico de los experimentos puede ahorrar tiempo y dinero en estos casos.

4 Herramientas utilizadas en el enfoque ALOHA

4.1 Simulación ABM y lazo SCADA

En esta sección se presenta un modelo genérico basado en agentes (de inglés ABM) para aplicaciones del ámbito industrial, al mismo tiempo se muestra el lazo SCADA creado entre el entorno virtual de agentes (VA_E), en este caso un entorno simulado basado en ABM y los componentes de la realidad o *shop-floor*.

Para ello, se muestra el ABM y el lazo de supervisión, control y adquisición de datos (SCADA), este último expresado mediante líneas gruesas con.

Los ABMs ayudan a describir sistemas altamente complejos dividiéndolo en entidades menos complejas, los agentes.

La mayoría de las aplicaciones requieren que su implementación se realice utilizando sistemas capaces de funcionar/adaptarse a la dinámica de los cambios, por ejemplo las variaciones en la demanda de productos, fabricación de nuevos productos, cambios en la organización de la planta (movimiento/agregar/quitar componentes), etc. Bajo este contexto, los sistemas industriales no son la excepción. Para poder hacer frente a ello, estos sistemas deberían transformarse en autónomos [43].

Los sistemas autónomos se caracterizan porque las entidades del mismo dicen a las otras entidades qué necesitan, pero no cómo deberían lograrlo. Incluso, dentro de las características de los sistemas autónomos está la capacidad de negarse a realizar algo por algún motivo que crea válido para su supervivencia sin afectar negativamente las metas globales del sistema.

Siguiendo este principio, un modelo ABM genérico se plantea en la Fig. 2 donde los agentes se dividen en dos grupos, los denotados como A_{ASR} que representan a agentes que hacen de interface con recursos específicos de la aplicación industrial en cuestión. Por ejemplo, en un sistema de producción podría ser un agente encargado de la planificación de la misma y en un sistema de transporte podría ser un agente responsable de la coordinación del tráfico. Estos

agentes pueden ser una interfaz que representa/modeliza a algún componente que existe en la realidad y que tiene contacto con el entorno mediante sensores/actuadores, o bien, simplemente pueden ser agentes específicamente diseñados/creados y modelizados para la aplicación en cuestión, como lo sería por ejemplo, el coordinador de tráfico o el planificador de los ejemplos anteriores.

El segundo grupo de agentes, pertenece a aquellos denominados ALOHA, este tipo de agentes tienen una arquitectura particular denominada ALOHA que a su vez representa un enfoque de cómo integrar componentes de la realidad en un agente. En este sentido, posee una naturaleza híbrida/heterogénea, dado que representa a un componente de la realidad y su modelo simultáneamente y, a diferencia del primer grupo de agentes (A_{ASR}), su razón de ser es la de 'embeber' componentes de la realidad dentro de una abstracción/arquitectura también llamada ALOHA. Además, este agente presenta la particularidad de que el componente simulado y el real se ejecutan en concurrencia. A modo de ejemplo, los componentes podrían ser AGVs en una aplicación relacionada con el transporte de objetos, o bien, podrían ser brazos robots que forman parte de una cadena de producción, etc.

Los agentes del ABM se comunican entre ellos utilizando un lenguaje en común denominado lenguaje de comunicación de agentes (del inglés *ACL*), donde en el caso de los agentes ALOHA es el nivel deliberativo/alto nivel (H) el que se comunica con el o los correspondientes miembros de A_{ASR}, de similar manera, los agentes ALOHA se comunican entre ellos también utilizando un ACL a través del correspondiente nivel $H_{1..n}$.

El modo de interacción es que los primeros (A_{ASR}), le dicen a los segundos (ALOHA), qué necesitan, pero no cómo llevar a cabo dicha asignación/solicitud/orden, etc. En otras palabras, la ejecución acontece de manera distribuida [44].

Por otra parte, en el caso de los modelos del bajo nivel o reactivo (L_V), que representan a los componentes de la realidad L_R, su interacción es con el entorno virtual A_E a través de sensores y actuadores virtuales.

En el caso de los módulos que integran SYM, su función es la de sincronizar el modelo con su contraparte real y será explicado en detalle en la sección 4.3.

Respecto a VA_E, se encuentra integrado por diferentes elementos, por un lado el ABM, descripto anteriormente, y por otro lado los elementos que actúan como entradas de configuración del ABM, los cuales pueden dividirse en dos grupos, un grupo representando por parámetros más estáticos y el otro por parámetros más dinámicos. En el caso de los estáticos, se seleccionan inicialmente y no pueden ser cambiados. Los dinámicos pueden variar su valor ya sea por interacción del usuario o de manera autónoma por decisión de algún agente autorizado a ello. Finalmente, VA_E también está compuesto por la interfaz usuario-máquina (HMI), que permite configurar los diferentes escenarios de ejecución, la visualización de la planta en tiempo real, etc.

Resumiendo, VA_E en el caso de esta tesis está representado por la simulación ABM que ha sido programada en Netlogo, la cual contiene la HMI, los parámetros de configuración y ejecución de la simulación, el modelo de cómputo basado en agentes, la arquitectura de control y el módulo de sincronización intra-agente para los agentes tipo ALOHA.

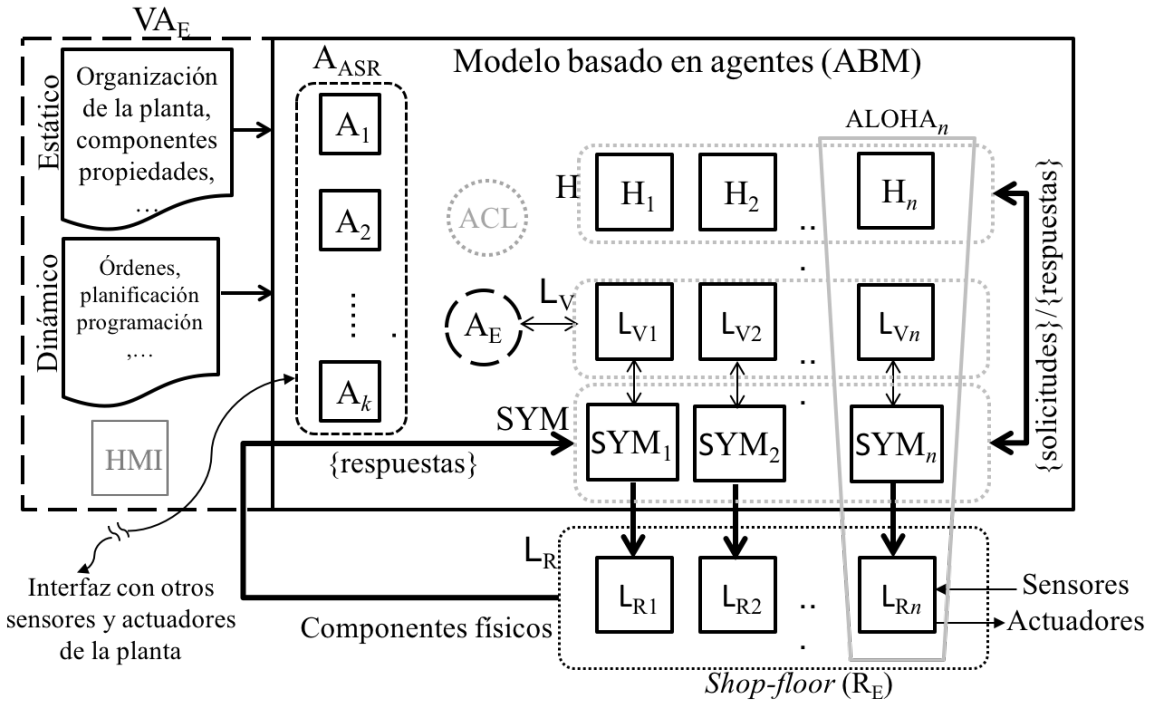


Fig. 2 Diagrama del ABM y del controlador basado en simulación ABM

Hasta el momento, se ha mostrado las características del modelo de cómputo implementado con el simulador ABM y las interrelaciones entre los diferentes agentes y parámetros de configuración, haciendo referencia mínima a su utilización como controlador.

El controlador basado en simulación ABM posee varios usos bien diferenciados: 1) Para validación funcional de la simulación, 2) para validación funcional del módulo SYM, 3) Para caracterización de la planta, 4) Para validación funcional de la aplicación final, lo que implica la ejecución de la simulación ABM como SCADA de la realidad, donde el modelo simulado se ejecuta en concurrencia con la misma.

Para el primer uso, se requiere básicamente que el nivel *H* y el nivel L_V estén implementados, con ello se puede comprobar que el modelo se comporta acorde a lo esperado en la realidad.

En el caso 2, donde se pretende validar *SYM*, se necesita tener validados *H*, L_V e implementar *SYM*. Cabe destacar que la implementación de SYM representa un módulo independiente, el cual luego de su desarrollo es verificado y validado, sin embargo, resulta conveniente que su validación sea realizada dentro del contexto del agente ALOHA. Validar SYM dentro del contexto ALOHA implica que este reciba las solicitudes de *H* y las distribuya a los niveles correspondientes del bajo nivel (L_V y L_R), en este caso dado que todavía L_R no se encuentra implementado, cada vez que se genera un evento desde L_V, se genera un evento 'fantasma/virtual' como si viniera desde L_R donde ambos son enviados a *SYM* y este discrimina si corresponde enviar la respuesta únicamente a *H* o bien tanto a *H* como a *L*, según sea el caso sin o con error.

En el tercer punto, donde el controlador se utiliza con motivo de caracterización, requiere que los dos casos anteriores hayan sido satisfactorios al igual que la implementación del nivel L_R. Caracterizar en este caso implica recibir valores de ciertas variables relacionadas con la ejecución del componente real (AGV, brazo robot, etc.), las mismas podrían ser velocidad, tiempo entre segmentos de un recorrido, nivel de baterías en función del uso en caso de AGVs, etc. Este proceso sirve para obtener parámetros de la planta de manera dinámica, los cuales pasan a formar parte o actúan como entradas para L_V.

Finalmente, el desafío de la validación funcional de la aplicación completa requiere que los tres casos anteriores hayan sido satisfactorios y además que la simulación sea capaz de

funcionar en concurrencia con la realidad. En este caso el controlador se ejecuta de manera distribuida para cada agente del tipo ALOHA. El lazo de control comienza por una solicitud enviada desde *H* hacia L_V y L_R, la cual se canaliza a través del módulo de sincronización (*SYM*). Las respuestas a la solicitud de *H* son enviadas de manera independiente desde L_V y L_R hacia *SYM*, las cuales se pretende que estén sincronizadas tanto en tiempo como en tipo de evento (mensaje); en dicho caso (sin error) las señales son enviadas a *H* donde, si corresponde, se envía una nueva solicitud. En caso de error, lo cual implica eventos de L_V y L_R no simultáneos y/o mensajes no iguales, se envía un mensaje de error desde el SYM hacia los niveles correspondientes.

Con todo, se ha explicado el modelo de cómputo y la arquitectura de control desde la simulación ABM que se utiliza para SCADA de la planta real. En la siguiente sección se presenta con mayor detalle la arquitectura del agente ALOHA, la cual, se ve que está relacionada con la arquitectura de control desde la simulación ABM a través de los módulos/componentes que se encuentran dentro del cono invertido truncado de la Fig. 2.

4.2 Arquitectura del agente ALOHA

Los agentes tipo ALOHA resultan clave en la arquitectura de control distribuida presentada en la Fig. 2. Estos agentes utilizan un enfoque dividido en tres niveles y una comunicación intra-agente entre el nivel superior (deliberativo) y el nivel inferior (reactivo) a través de un módulo de sincronización (SYM) entre el modelo simulado y su contraparte de la realidad.

En la Fig. 3 se puede observar la arquitectura interna para un agente genérico tipo ALOHA denominado *i*, la misma se encuentra dividida en tres niveles, siendo el alto nivel (deliberativo) representado por H_i, el bajo nivel constituido por L_{Vi} y L_{Ri} siendo el modelo y la contraparte real respectivamente. La capa intermedia entre estos dos niveles, llamada SYM, es la encargada de la sincronización del modelo (L_{Vi}) con su contraparte real (L_{Ri}).

H_i interactúa a través de mensajes utilizando un lenguaje de comunicación de agentes (del inglés ACL) FIPA compatible con el alto nivel de otros agentes ya sean del tipo ALOHA o A_{ASR}.

Para el caso del bajo nivel, L_{Vi} interactúa a través de sensores y actuadores con el entorno virtual de agentes A_E (una representación de la realidad en el simulador) y análogamente L_{Ri} (embebido en el componente CO_i) con el entorno real R_E.

En color gris se pueden ver las comunicaciones que tienen lugar dentro del entorno simulado, mientras que en color negro, se pueden ver las interacciones con la realidad.

La capa intermedia hace de interfaz entre H_i y L_i suministrando todos los servicios para comunicarlas de manera protegida y segura. Esto incluye procedimientos para enviar solicitudes/comandos a la capa interior y gestionar las respuestas de la misma.

De esta manera, tanto L_{Vi} y L_{Ri} son controlados por comandos enviados desde H_i, dichos comandos dependen de diferentes factores, como pueden ser los mensajes de otros agentes ALOHA o A_{ASR}, de la información global de A_E, de las respuestas de L_{Ri} y también de las diferencias entre las respuestas de L_{Vi} y L_{Ri} que se gestionan a través de SYM.

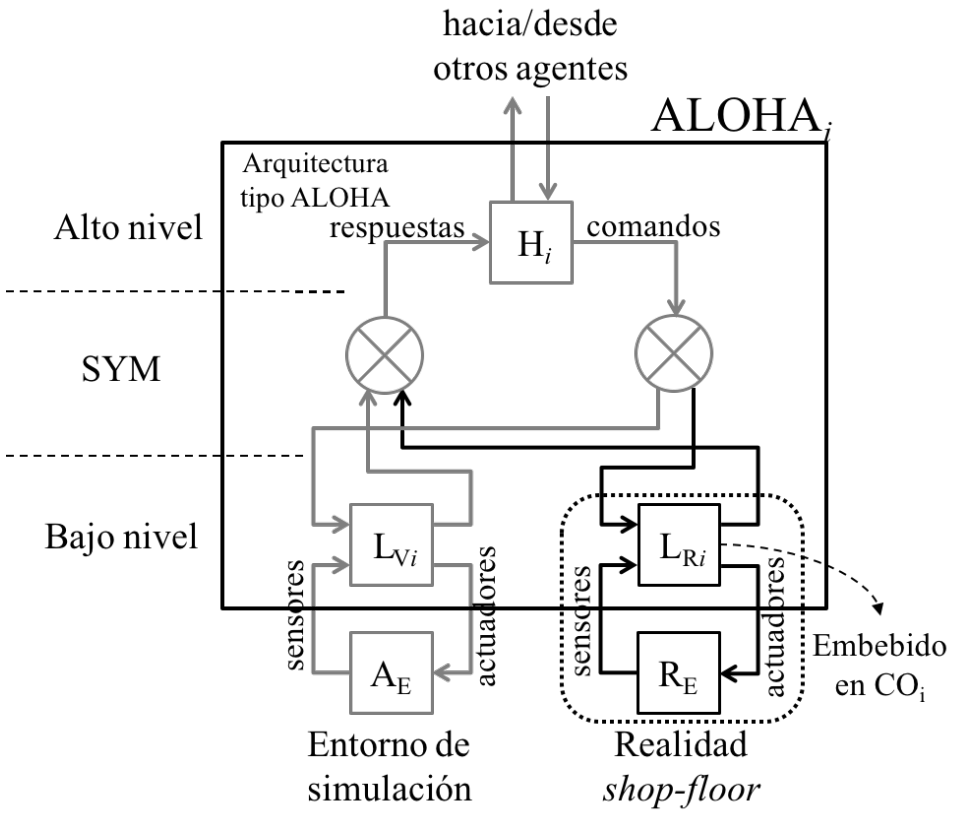


Fig. 3 Arquitectura del agente ALOHA

La arquitectura aquí presentada permite convertir en agente a cualquier componente de la realidad (*CO*), sincronizar el modelo con su contraparte real (para mantener una vista simbólica de la realidad en el entorno simulado) y realizar SCADA en tiempo real de cada agente ALOHA.

4.3 El módulo sincronizador (SYM)

El sincronizador es un recurso clave de la arquitectura del agente ALOHA, ya que provee los medios básicos para tener una simulación *online* y concomitante con la realidad.

El sincronizador es un *middleware* interno a cada agente ubicado entre los niveles *H* y *L*. El mismo se encarga de gestionar los mensajes entre ambos, tanto los que van desde *L* hacia *H* y viceversa. Dado que el sincronizador gestiona mensajes del tipo *time-tagged*, se hará referencia a ellos simplemente como eventos.

Hay dos clases de eventos: los causados por solicitudes desde H_i y otros diferentes a estos. Los primeros tienen que ser sincronizados en *hard-real time* mientras que los otros permiten algún tipo de falta de coincidencia o desajuste entre la realidad y la simulación por ende son los que se denominan como sincronización *soft-real time*.

Los eventos de sincronización tipo *hard-real time* (HSEs), incluyen las solicitudes emitidas por H_i y las correspondientes respuestas desde $(L_V, L_R)_i$, los cuales se esperan que ocurran simultáneamente.

Los eventos de sincronización tipo *soft-real time* (*SSEs*) incluyen mensajes del tipo informativos desde $(L_V, L_R)_i$ y que se refieren a condiciones que son gestionadas autónomamente por el bajo nivel (*L*). Por ejemplo, detectar un obstáculo o informar sobre la batería baja son situaciones, que son manejadas de manera autónoma/localmente por la correspondiente entidad de *L*, lo que significa que no requieren de atención inmediata por parte del *H*. Es importante destacar, que el *L* se ejecuta de manera sincronizada para tener una representación virtual precisa de la realidad y, en el caso de realidad mixta, tener a la realidad interactuando en conjunto con agentes sin contraparte real (únicamente simulados).

Sincronizar la simulación con la realidad para el agente ALOHA$_i$ significa 1) mantener los mensajes de la simulación esperando por los mensajes de su contraparte real y 2) avanzar la simulación para disparar el evento correspondiente al evento de la realidad. El sincronizador gestiona estos eventos como así también aquellos relacionados con errores, por ejemplo, cuando un evento no puede encontrar su contraparte dentro de una brecha de tiempo.

Los eventos son denotados como *h* o *s*, dependiendo si son del tipo HSE o SSE. Se utilizan subíndices para indicar el origen del evento y un asterisco como superíndice que corresponde a los errores de sincronización. Los subíndices pueden ser *H*, *S*, *R* o *V* asociados a los eventos que vienen del alto nivel, del sincronizador, del componente real o del simulado, respectivamente.

Por ejemplo, h_H corresponde a un HSE que proviene de H, h_S^*, para un HSE proveniente del sincronizador y reportando un error, y s_R y s_V, para SSE con origen en el componente real y el simulado, respectivamente.

La Fig. 4 muestra un diagrama de bloques para el sincronizador donde se ven todos los posibles eventos que puede gestionar, tanto cuando hay errores y en el caso sin error.

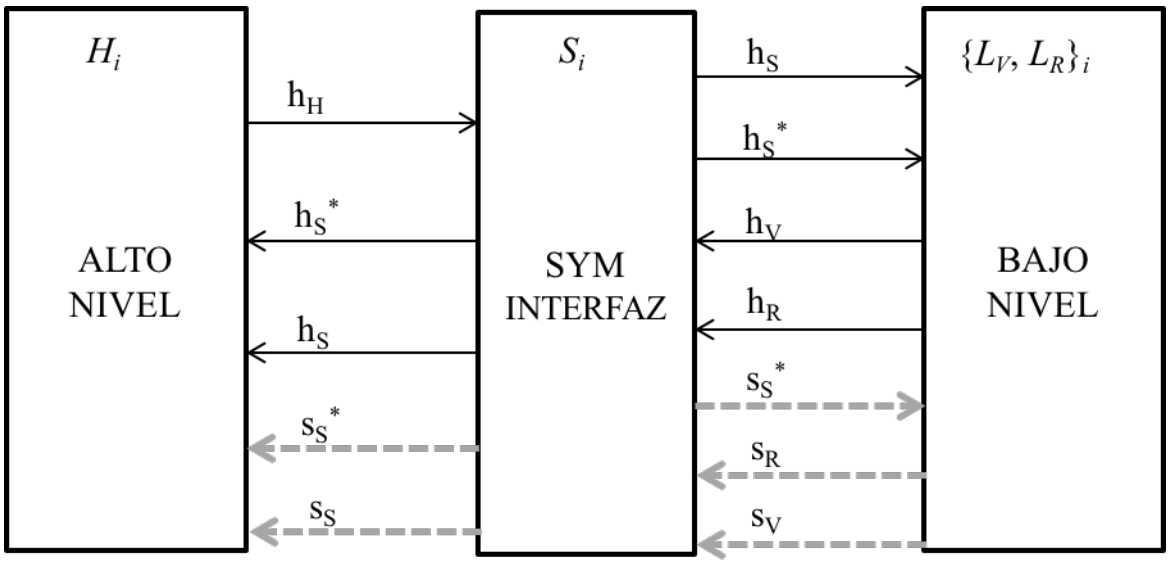


Fig. 4 Diagrama de bloques del sincronizador para un agente ALOHA$_i$

Hay que tener en cuenta, que para cada par de eventos de L el sincronizador debe producir un evento equivalente, por ejemplo para cualquier par h_V y h_R que responden a un h_H causado por el alto nivel, tiene que haber un h_S hacia H, en caso de que no haya error. Si algo falla, el evento de error del sincronizador más apropiado será el h_S^*, que se envía a H y eventualmente a L.

El protocolo de comunicación para el caso libre de error, comienza por una solicitud enviada por h_H hacia el sincronizador, quién a su turno envía el correspondiente h_S a L. Después de cada h_H, el sincronizador espera la recepción de los eventos correspondientes de h_V y h_R. Si ambos ocurren en el mismo ciclo de monitoreo y control y además los mensajes son iguales, el sincronizador envía la respuesta h_S a H, con el contenido del mensaje de L. Si los mensajes no son los mismos, el sincronizador emite un evento de error h_S^* hacia ambos niveles (H y L). Si los eventos de L no ocurren simultáneamente, el sincronizador espera al evento de h_R o bien si el evento de h_R ha llegado primero, la simulación dispara un evento para que el sincronizador reciba el h_V y se sincronice con la realidad.

La detección de obstáculos y otros tipos de SSEs suceden a nivel L y ayudan a sincronizar la representación virtual con la realidad, aunque admiten algún tipo de desajuste y es una de las causas por la que son considerados tipo *soft*. En este caso, el sincronizador trata de emparejar cada s_R con su correspondiente s_V, usando una estrategia similar a la de h_V y h_R, sin embargo, estos poseen una tolerancia temporal entre ellos para dar tiempo a su ocurrencia. Nuevamente, cuando los mensajes son iguales se envía un s_S a H y en el caso de error, se envía s_S^* tanto a H como a L.

El envío de s_S^* a L_{Vi} si no ha ocurrido ningún s_V permite a L_{Vi} percibir eventos no esperados de la realidad y de la misma manera, el envío de s_S^* a L_{Ri} si no sucedió ningún s_R, permite a L_{Ri} percibir estímulos generados por simulación.

Internamente el sincronizador clasifica los eventos que le llegan en tres diferentes estados, los cuales son válidos tanto para HSEs y SSEs, los cuales se llaman:

In-time cuando los eventos de la realidad y la simulación llegan al mismo tiempo o dentro del margen de tolerancia.

Ahead cuando los eventos de la realidad llegan primeros que los eventos de la simulación. Este es el peor caso en términos de sincronización y debe ser evitado tanto como sea posible, dado que la realidad no puede esperar a la simulación para que ambos estén sincronizados.

Behind cuando los eventos de la simulación llegan primero que los de la realidad y la simulación tiene que esperar a la realidad, para luego, si corresponde, actualizar parámetros al nuevo comportamiento del componente de la realidad.

Dado que el sincronizador es independiente de la aplicación, se necesita incluir en las tramas de datos de los eventos (Fig. 5) algunos parámetros que ayuden a resolver problemas de tiempo o de contenido de mensajes. Estos parámetros se explican a continuación.

T_{out} es un *timeout* para h_R con respecto a un h_S que ha sido causado por un h_H. Si h_R no ocurre dentro de este período de tiempo, un evento de error h_S^* es enviado a ambos niveles (H y L). En este caso h_S^* tiene un mensaje que contiene el mensaje de *timeout* del h_H inicial. Este mecanismo esboza el comportamiento llamado *event discovery method* (EDM) del sincronizador.

I_{max} es el número máximo de ejecuciones del ABM para causar que la simulación dispare un evento de tipo h_V que corresponde a un evento previo h_R. En el caso que este *immediately synchronization method* (ISM) falle (por ejemplo el sincronizador no recibió el h_V antes de I_{max}), un h_S^* con un mensaje predefinido es enviado a H y L.

Los SSEs pueden ocurrir temporalmente desfasados pero dentro de un período menor a T_{tol}, por ende, T_{tol} es la tolerancia temporal umbral antes de comenzar un EDM para s_V o s_R.

T_{outs} es el timeout para SSEs durante la ejecución de un EDM para s_V o s_R.

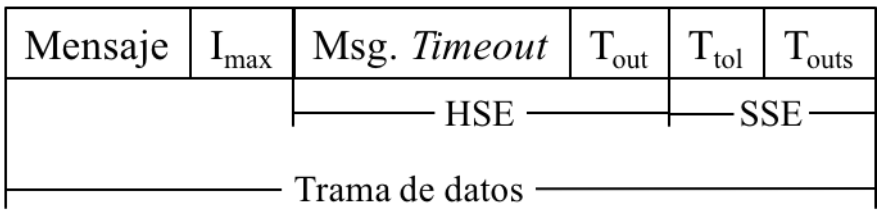


Fig. 5 Formato de la trama de los eventos

En la Fig. 6 se intenta explicar el funcionamiento de la sincronización entre el entorno simulado y la realidad. Para ello, se utilizan eventos genéricos de la realidad denominados *e* y eventos desde la simulación llamados *e*' los cuales representan diferentes situaciones que podrían ocurrir en el transcurso de un tiempo virtual (VT) para la simulación y un tiempo real (RT) perteneciente a la realidad. Cabe destacar, que una adecuada caracterización de la realidad y un lazo de control suficientemente rápido, deberían llevar a que VT sea cercano a RT. En este sentido, dado que VT es cercano pero no igual a RT, en general existen diferencias entre ambos, las cuales terminan disparando eventos distintos al de *in-time*. En la Fig. 6 se muestra cómo son gestionadas las discrepancias entre los eventos VT y RT, representando de manera gráfica lo explicado con anterioridad en esta sección.

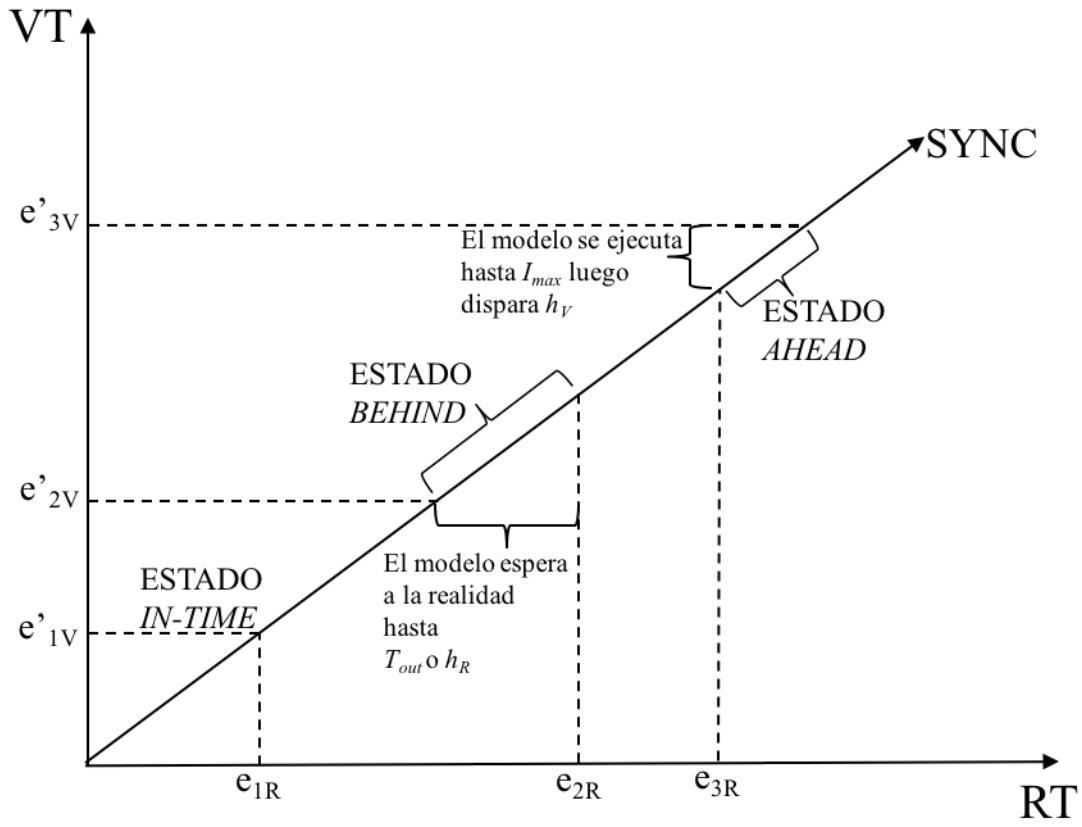


Fig. 6 Ejemplo de sincronización para HSEs (análogo para SSEs)

Como se puede observar, en la línea de sincronización el estado *behind* sucede cuando un evento de la simulación (h_V) llega primero, en ese caso, el modelo espera a la realidad un tiempo T_{out} (este caso el sincronizador gestiona un error) o bien hasta que el correspondiente h_R con igual mensaje que h_V llega dentro de la brecha de tiempo T_{out} (en este caso se resuelve la sincronización sin error). Para el estado *ahead,* el evento de la realidad (h_R) ocurre primero, en este caso la simulación ejecuta un número máximo de iteraciones I_{max} y en caso de llegar a producir el evento h_V dentro de ellas, se resuelve la sincronización sin error, caso contrario, el sincronizador deber gestionar un error enviando mensajes tanto a *H* como a *L*.

A través de la Fig. 6 se puede observar cómo la capa de sincronización intenta mantener la simulación al mismo ritmo que la realidad y cuando esto no ocurre y los márgenes de tolerancia tanto de T_{out} e I_{max} son desbordados, se puede ver, cómo el sincronizador provee a cada nivel de los eventos pertinentes y así cada uno puede actuar en consecuencia.

Hasta el momento se ha expresado al sincronizador como un módulo, cuya función es sincronizar simulación con realidad, a continuación se mostrará cómo está constituido el mismo.

El módulo de sincronización se compone de dos máquinas de estados finitos (del inglés FSM), una para la gestión de HSEs y otra para la gestión de SSEs.

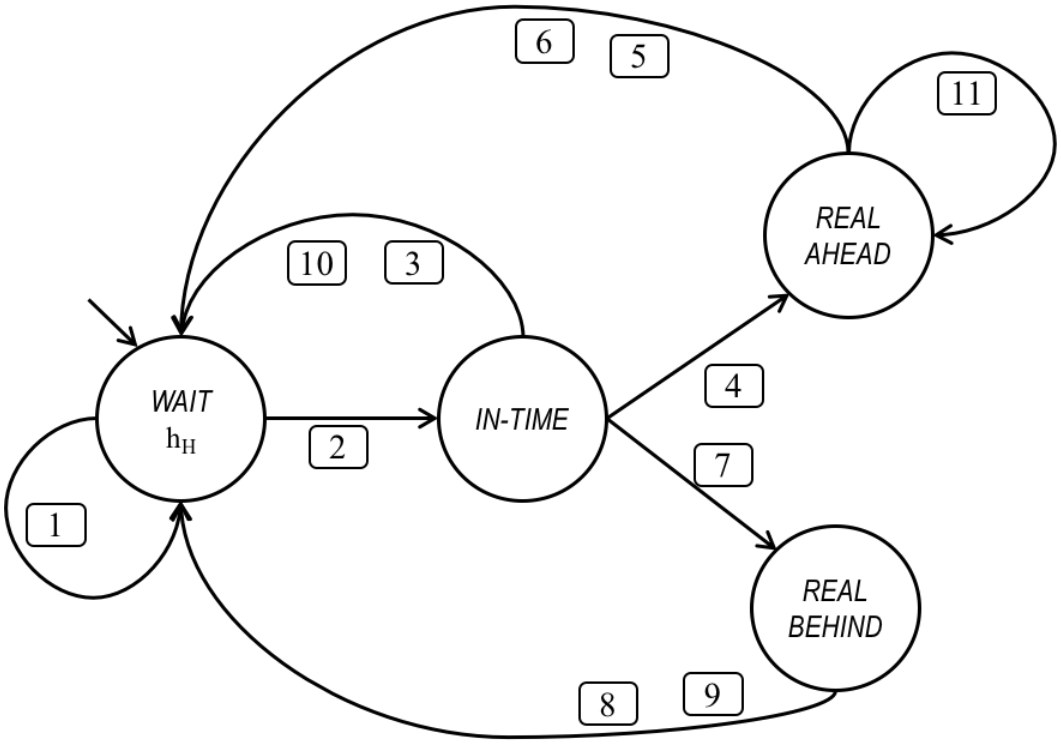
El comportamiento del sincronizador según HSEs es modelado por una FSM de cuatro estados, los cuales se pueden ver en la Fig. 6 y están descriptos por $S = \{WAIT, BEHIND, IN\text{-}TIME, AHEAD\}$, con un alfabeto de entrada $I = \{h_R, h_V, h_H, timeout, mismatch\}$ y un alfabeto de salida $O = \{h_S, h_S*\}$. Donde h_S contiene una copia de los eventos de entrada como respuesta de que no hubo error y h_S* contiene eventos con mensajes específicos para indicar diferentes errores. A continuación se explica lo esencial de la Fig. 6, ya que en ella se pueden observar los demás casos.

En el estado *WAIT*, se espera un h_H, cualquier otro evento tanto de la realidad como de la simulación ($h_R + h_V$) causará un mensaje de error h_S* que se enviará hacia ambos niveles (*H* + *L*), según se observa en (1).

Cuando un h_H es recibido (transición (2)) y ningún otro evento de entrada (el apóstrofe significa complemento) está presente al mismo tiempo, el estado se mueve a *IN-TIME* y envía una copia del mensaje h_H hacia L y comienza a incrementarse un temporizador que actúa como *timeout* o mejor dicho tipo perro guardián (*watchdog timer*). En este estado, el caso ideal sería que los eventos h_R y h_V lleguen al mismo tiempo según se representa en (10) y que los mensajes de ambos sean equivalentes. Pero, es posible que uno llegue antes que el otro, por lo tanto, si h_R llega primero, el estado cambia a *REAL-AHEAD* (transición (4)) y si el evento h_V ocurre antes que el h_R la transición que se produce es la (7) siendo el nuevo estado *REAL-BEHIND*. En el caso que ningún evento (h_R *o* h_V) llegue antes que se exceda el *watchdog timer* o llegue un nuevo h_H, entonces se responde con un evento de error y se retorna al estado *WAIT*.

Al realizar la transición al estado *REAL-AHEAD*, se resetea el contador de iteraciones de simulación y los mensajes son enviados de tal manera que la simulación avance hasta que el evento pendiente de la simulación h_V es recibido o alguna condición de error es disparada.

En el estado *REAL-BEHIND*, el sincronizador espera por el evento correspondiente de la realidad h_R o hasta que algún error (ya sea *timeout* o un nuevo h_H) ocurre.



1 $h_R + h_V / h_S$*
2 $h_H\ h'_V\ h'_R / t_0 = t, h_S$
3 $(h_H + timeout) / h_S$*
4 $h'_H\ h'_V timeout'\ h_R / k = 0$
5 $h_H + mismatch + h_V\ (Msg(h_V)=Msg(h_R))' / h_S$*
6 $h'_H\ mismatch'\ h_V\ (M(h_V)=M(h_R)) / h_S$
7 $h'_H\ h'_R timeout'\ h_V$ / -
8 $h'_H\ timeout'\ h_R\ (Msg(h_R)=Msg(h_V)) / h_S$
9 $h_H + timeout + h_R\ (Msg(h_R)=Msg(h_V))' / h_S$*
10 $h'_H timeout' h_V\ h_R / h_S$
11 $h'_H\ mismatch' h'_V / k = k + 1$

Fig. 7 FSM contenida en el sincronizador para la gestión de HSEs

La FSM de la Fig. 8 forma parte del módulo de sincronización y su función es la de gestionar los denominados SSEs. Para ello se define como alfabeto de entrada a $I = \{s_V, s_R, h_X\}$, donde s_V y s_R simbolizan los eventos que provienen de la simulación y de la realidad respectivamente, mientras que h_X determina que cualquier evento HSE es ignorado por esta máquina de estado y procesado por la de la Fig. 7. El alfabeto de salida está representado por O = $\{s_S$*, $s_S\}$, representando la ocurrencia de error y el caso de no error respectivamente. En cuanto a los

estados de la máquina se pueden diferenciar tres estados globales (líneas punteadas) siendo análogos a los de HSE representados por *S* = {*IN-TIME*, *REAL-AHEAD*, *REAL-BEHIND* }, sin embargo, para esta FSM hacen falta estados intermedios tanto para *REAL-AHEAD* como para *REAL-BEHIND*, que permitan la correcta gestión de los eventos tipo SSEs. En el estado *NO_WAIT*, es la única posibilidad para que, simulación y realidad sean concurrentes (*IN-TIME*), tal como se puede observar en (1) y, a diferencia del estado inicial para la FSM de HSEs que esperaba un evento de *H*, en este caso solo se esperan eventos con origen en *L*.

La transición al estado *REAL-AHEAD* (específicamente *WAIT_VIRTUAL*) sucede en (2), cuando se recibe primero un evento s_R (sin recibir un s_V simultáneamente), en ese momento se inicializa un temporizador que provee una tolerancia temporal (T_{tol}) a la espera del evento de s_V, si el evento llega antes de exceder T_{tol}, se produce la transición (8), caso contrario, se transita mediante (4) al estado *WAIT2_VIRTUAL* donde se ejecuta el EDM durante un tiempo T_{outs}, esta segunda espera permite regresar al estado *NO_WAIT* a través de (6) ya sea con un error (s_S*) o sin error (s_S). El estado *WAIT_X*, donde X = *REAL*/*VIRTUAL*, pretende dar la oportunidad para que la sincronización ocurra de manera natural, mientras que el estado *WAIT2_X*, tiene como intención descubrir el evento y en consecuencia generarlo oportunamente.

La descripción provista para el párrafo anterior es análoga cuando el evento que lleva primero es un s_V, el cual se inicia a través de la transición (3).

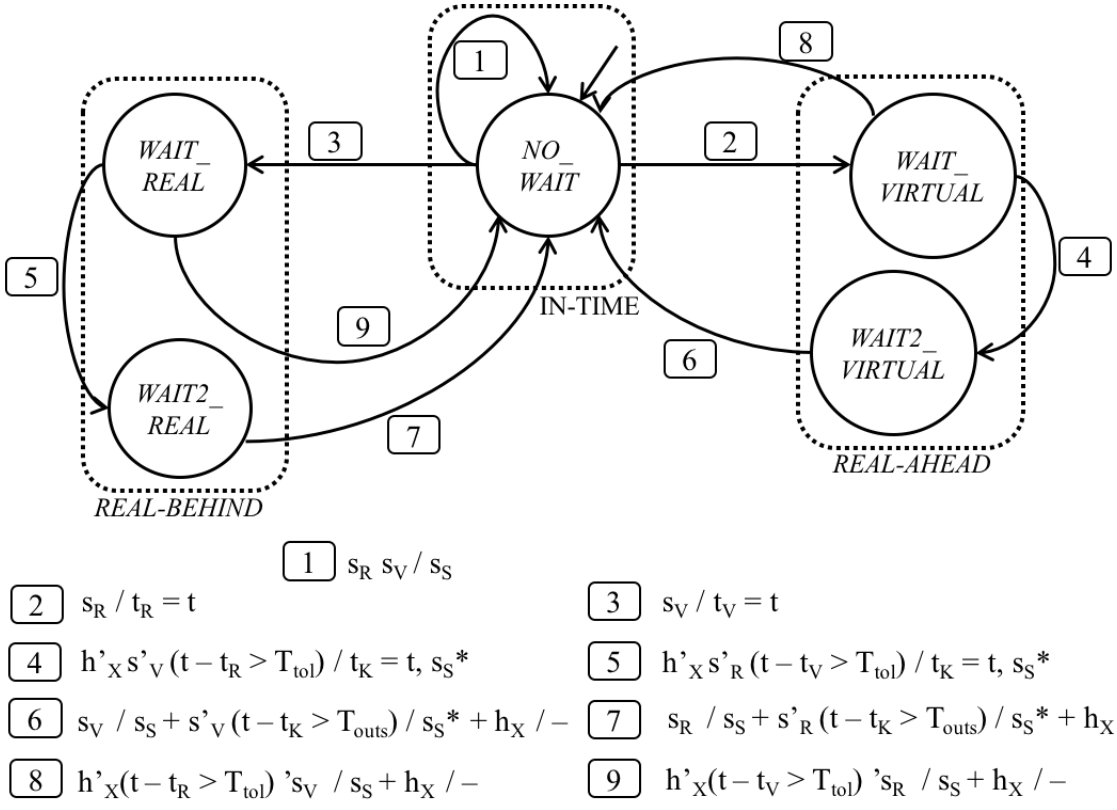


Fig. 8 FSM contenida en el sincronizador para la gestión de SSEs

Hasta el momento se ha citado durante esta sección la teoría correspondiente a la sincronización entre simulación y realidad para los agentes tipo ALOHA, a continuación se pretende explicar de manera más descriptiva el funcionamiento de la sincronización y para ello se utilizará un ejemplo que integra tanto HSEs como SSE para AGVs que han sido convertidos en agentes tipo ALOHA.

La Fig. 9 muestra una escena con dos agentes tipo ALOHA, donde el denominado $ALOHA_1$ se mueve por detrás del $ALOHA_0$ a lo largo de un mismo segmento (una ruta entre dos marcas denotadas como X). Sobre la línea superior de tiempo se muestran los AGVs reales, mientras que en la inferior se representan las contrapartes virtuales. Tanto los AGVs reales como los virtuales están compuestos por sensores que permiten detectar marcas y sensores de distancia (dentro de un rango *d*) para detectar obstáculos, estos últimos representados como las semicircunferencias con puntos delante de los vehículos.

En color gris, hacia la izquierda se puede ver el estado inicial de los vehículos antes de comenzar la trayectoria. Al producirse la detección de la marca (X) cada agente recibe el correspondiente mensaje h_H de su alto nivel (H), lo que permite que se muevan hacia la marca que está más hacia la derecha.

Hacia la derecha, con los vehículos en color negro, se ilustra un caso de HSEs para el agente $ALOHA_0$ mientras que el agente $ALOHA_1$ representa un caso de SSEs.

En el diagrama de secuencias basado en nomenclatura UML de la Fig. 10, se ilustran casos de cómo el sincronizador gestiona los mensajes de los eventos que llegan tanto para las situaciones con o sin error. Para poder explicar con mayor claridad, se agrega una nomenclatura propia a las que forman parte de un diagrama de secuencia tradicional.

Los mensajes/eventos aparecen dentro de círculos numerados tanto en la Fig. 9 como en la Fig. 10, los números con apóstrofe representan error mientras que aquellos que aparecen sin apóstrofe denotan que no ha habido error. Las líneas grises dentro de lo que representa el sincronizador indican el estado *in-time*. Cabe destacar que solo para una mejor organización del diagrama, el sincronizador aparece fuera de los agentes ALOHA, dado que es interno a cada agente.

El ejemplo mostrado se basa en experimentos realizados en el laboratorio con la plataforma desarrollada en esta tesis y los números en la Fig. 10 indican el orden de ocurrencia de los eventos, los cuales se detallan a continuación.

(1) Desde H_0 se emite el correspondiente h_H y en consecuencia el sincronizador (SYM), envía el correspondiente h_S hacia L y cambia de estado para indicar que la simulación va in-time con la realidad.

(2) Mientras $ALOHA_0$ avanza, $ALOHA_1$ encuentra una marca, por ende H_1 emite el correspondiente h_H, para que luego el sincronizador emitir el h_S hacia L, ahora tanto $ALOHA_0$ como $ALOHA_1$ avanzan hacia la marca que se encuentra más hacia la derecha.

(3) La contraparte virtual de $ALOHA_0$ (V_0) encuentra una marca y envía el correspondiente h_V hacia SYM, en consecuencia, SYM detecta que el correspondiente h_R no llega simultáneamente; por ende se lanza el EDM para esperar el arribo de h_R. En este caso,

la diferencia entre la parte real y la virtual es causada por un área resbaladiza (una capa fina de aceite en el suelo).

(4) Mientras el EDM de (3) se está ejecutando, un obstáculo es encontrado por la parte real de *ALOHA1*, por lo tanto SYM recibe el correspondiente s_R, el sincronizador detecta que s_V no ha llegado simultáneamente con el s_R e inicia el EDM.

(5) El EDM de (3) falla dado que h_R no llega dentro del tiempo de tolerancia establecido y envía el correspondiente error h_S*.

(6) El EDM de (4) también falla dado que el s_V no se manifiesta dentro de los márgenes de tolerancia temporal, por ende el SYM decide lanzar el ISM el cuál fuerza un obstáculo virtual (el hexágono), de esta manera, dado que el SYM recibe ambos mensajes (s_R y s_V) envía hacia H_1 el correspondiente s_S para indicar que se ha podido restablecer del fallo y que todo está correcto.

(7) Paralelamente, R_0 avanza luego de recibir (1'), lo que dispara inmediatamente (3') y se libera el obstáculo visto por R_1, lo que causa que se envíen los correspondientes s_R y s_V y, como consecuencia de esto SYM envía s_S.

(8) Antes de que el EDM lanzado por (3') finalice, el esperado h_R llega por ende SYM envía h_S para comunicar que todo está correcto para el lazo de HSE que se inició previamente por H_0.

(9) Dado que tanto R_0 como su contraparte virtual V_0 encuentran la marca más hacia la derecha (X), $ALOHA_0$ envía desde H_0 un nuevo h_H.

(10) $ALOHA_0$ continua avanzando y a su turno, la parte virtual y real de $ALOHA_1$ llegan al mismo tiempo a la marca más hacia la derecho (en este caso R_1 no se ve afectado por el área resbaladiza), en consecuencia, SYM envía el h_S para el previo h_H emitido en (2).

(11) Dado que $ALOHA_1$ ha detectado las marcas, H_1 envía un nuevo h_H.

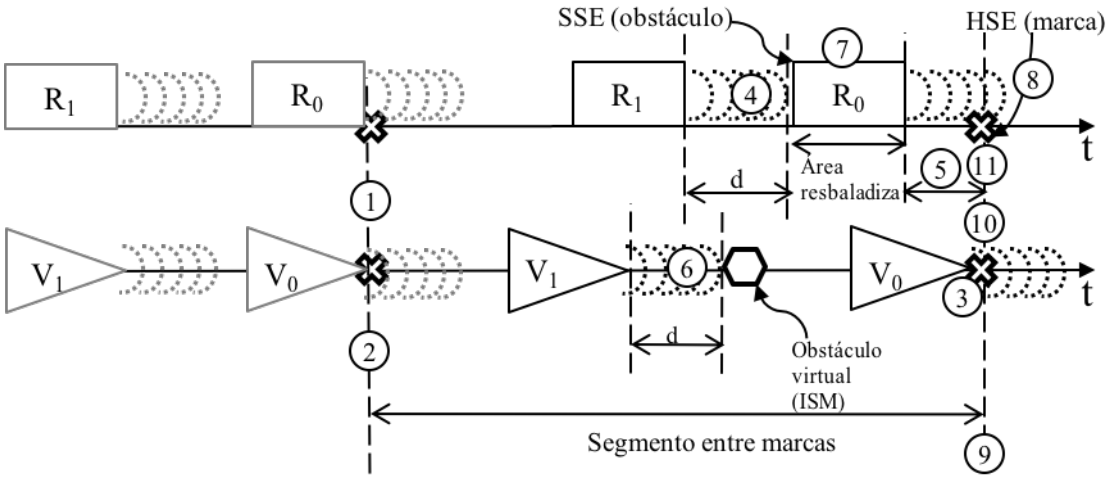


Fig. 9 Escena con dos AGVs (reales y virtuales) con HSEs y SSEs

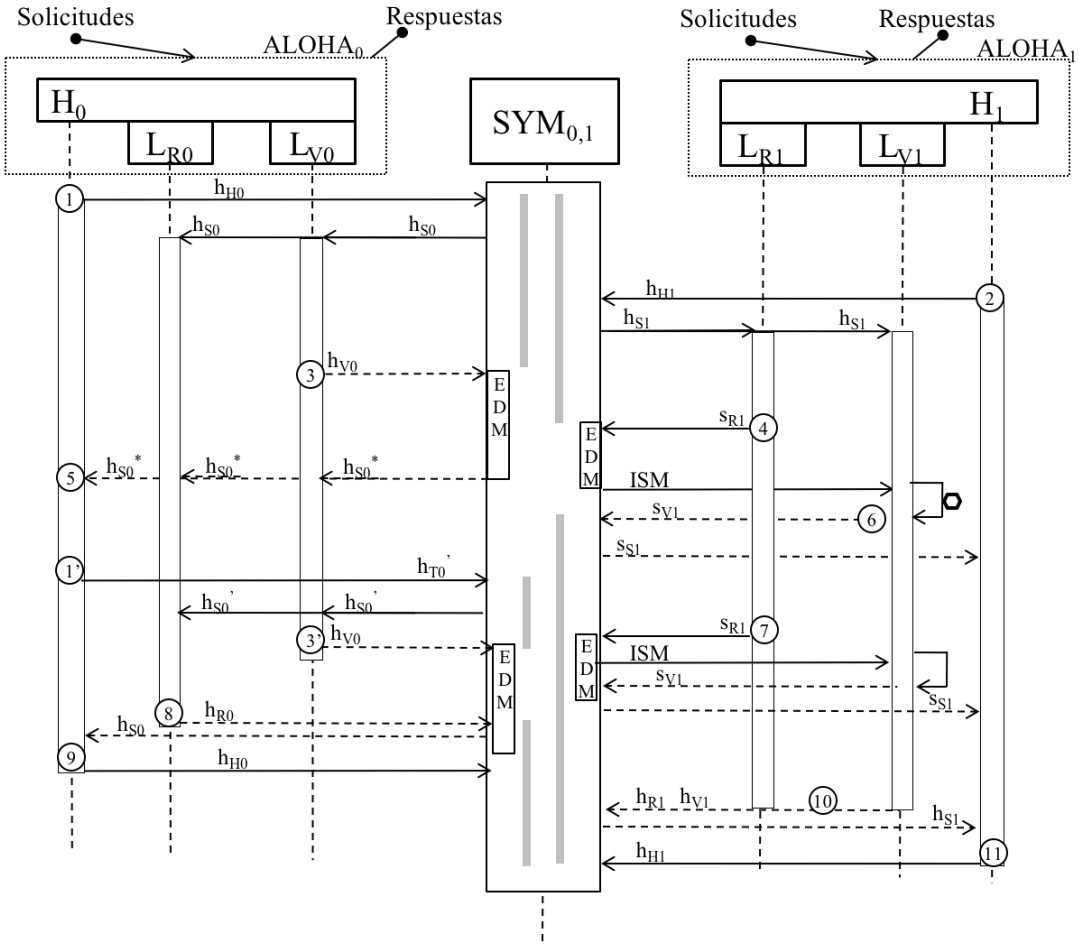


Fig. 10 Comunicaciones intra-agente para HSEs y SSEs de la Fig. 9

4.4 Metodología ALOHA

En esta sección se explica por medio de la Fig. 11 cómo se pueden llevar a cabo sistemas industriales desde el diseño hasta la implementación de los mismos, donde también se incluyen aquellos agentes denominados ALOHA.

El proceso de diseño comienza con la creación del ABM del sistema, lo que incluye también la simulación de la parte física (sistema real). En esta parte se crea el entorno de agentes (A_E), los agentes específicos de aplicación (A_{ASR}) y los agentes tipo ALOHA. Este último tipo de agente se organiza en dos módulos: el primero que se ocupa de las funciones de control de alto nivel (H) y las comunicaciones con otros agentes, mientras que el bajo nivel (L) es responsable del control local a nivel de campo siendo a nivel de simulación L_V.

Después de la verificación y validación funcional a nivel de simulación, la parte de bajo nivel L_V debe ser embebida en los componentes de la realidad (AGVs, brazos robots, etc.) representados como L_R.

En este punto, los L_V podrían contener valores de parámetros teóricos, nominales, etc., tales como máxima velocidad, capacidad de carga de baterías, etc.; los cuales podrían diferir de los parámetros reales. Afortunadamente, el módulo de sincronización permite operar el sistema aún con parámetros erróneos, dado que la tolerancia inicial del sincronizador es amplia. Luego de unas pocas ejecuciones, los parámetros del sistema pueden ser sintonizados finamente, y luego los parámetros de tolerancia del sincronizador son ajustados a valores operativos finales. Sin embargo aunque no es impuesta por esta metodología, es importante que los agentes a nivel de sistema incluyan un módulo para la identificación adaptativa de parámetros para poder realizar estas operaciones *on-line*.

Finalmente, la implementación de SCADA de la realidad desde la simulación queda satisfecha de manera directa. Cabe destacar, que una unión típica entre los componentes de la arquitectura y el modelo de los mismos requeriría que el ABM se ejecute en un ordenador y, que el bajo nivel virtual del agente sea replicado o embebido en los componentes de la realidad, lo que hace también necesario que se provea la infraestructura de comunicaciones adecuada, ya sea inalámbrica o cableada y que además la plataforma de test sea la misma que la de la realidad

o muy parecida a ella. Por consiguiente, este flujo de diseño no solo permite implementar un prototipo más rápido, sino también, un despliegue ágil del sistema SCADA.

En el flujo de diseño propuesto se utilizan un conjunto reducido de herramientas, una para el modelado y la simulación y otra para el desarrollo de la parte embebida del agente. Típicamente, Java, C y sus variantes pueden utilizarse y ya han sido utilizados en otros trabajos, ya sea en sus versiones basadas en orientación a objetos como orientadas a agentes. Sin embargo, en esta tesis se ha aplicado un enfoque más agresivo, desde el punto de vista de cumplir especificaciones de tiempo real, al utilizar Netlogo [113], ya que el mismo está basado en Java y Scala, su programación se realiza en un lenguaje del estilo logo pero diseñado para agentes y los programas son traducidos a Java en tiempo de ejecución, por ende, la misma agrega un retardo en tiempo de ejecución lo que dificulta y por ende vuelve más agresiva la factibilidad de SCADA en tiempo real. Sin embargo al programar en Netlogo se pueden escribir instrucciones con complejas y con un alto nivel de abstracción lo que hace más fácil y rápida la programación de ABM complejos.

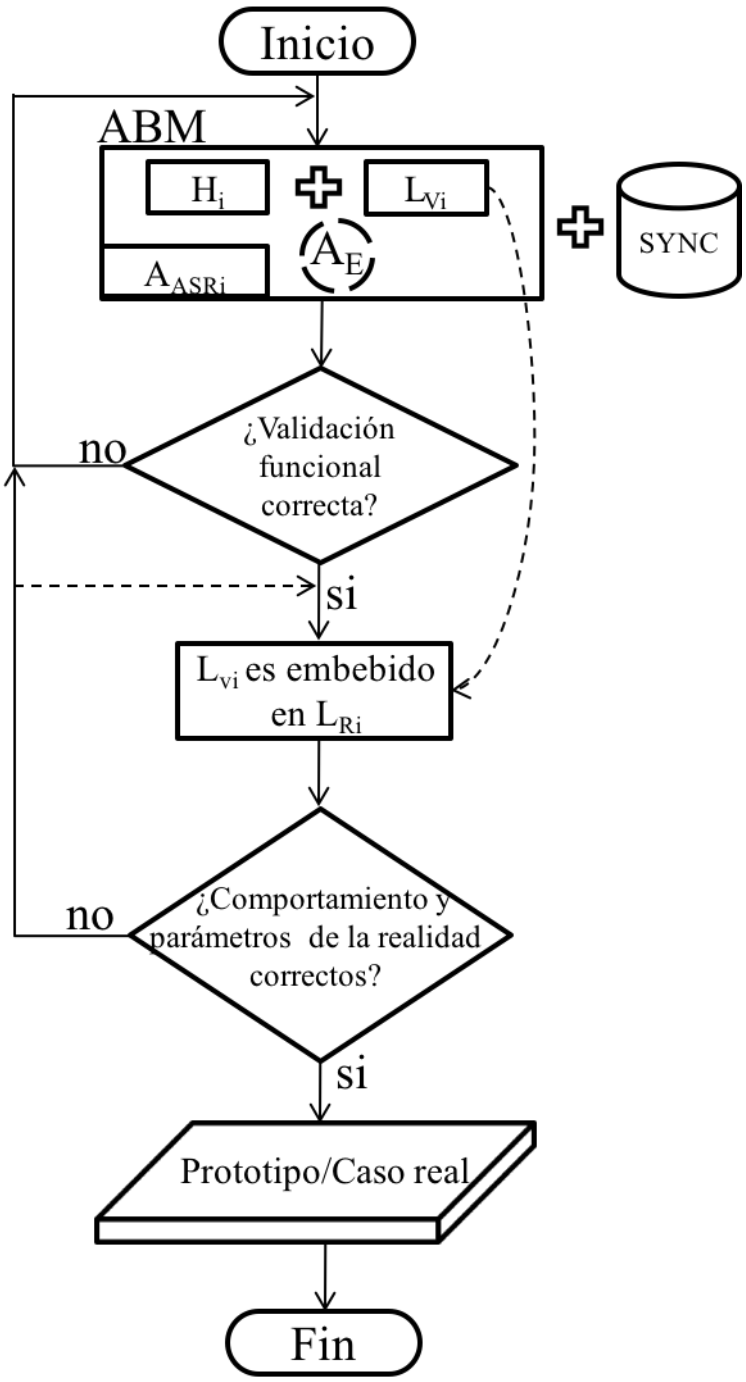


Fig. 11 Metodología desde el diseño hasta la implementación

5 Prototipos y validación en el área industrial

En esta sección se presentan dos prototipos que muestran la factibilidad de utilizar simuladores ABM para SCADA de aplicaciones industriales. Ambos han sido desplegados utilizando el enfoque ALOHA y las herramientas desarrolladas en el marco de esta tesis. El objetivo de esta sección también es de servir como validador de la hipótesis planteada en la introducción. El primer prototipo se relaciona con una aplicación de transporte de muestras utilizando AGVs en laboratorios de análisis clínicos automatizados. El segundo prototipo representa una celda de producción, donde un brazo robot neumático es encargado de mover objetos hacia la siguiente etapa del proceso. Cabe destacar que la aplicabilidad de la propuesta de esta tesis no está limitada únicamente a los prototipos utilizados, sino que es extensible a cualquier tipo de aplicación industrial tanto de logística como de producción, siempre que los requerimientos de la aplicación puedan ser cumplidos con el *framework* desarrollado en esta tesis.

5.1 Robótica móvil (AGVs)

Los robots de guiado autónomo o en inglés AGVs, son cada vez más utilizados tanto para el transporte interno (almacenes, fábricas, etc.) como para el externo (puertos de carga/descarga, transporte de personas en aeropuertos, etc.). Este proceso de automatización contribuye a que las tareas de transporte se realicen de manera más eficiente [114]. En la actualidad esta clase de robots permite mejorar la logística de organizaciones tanto para el transporte de objetos como de personas, sin embargo todavía existen muchos desafíos que tienen que enfrentarse [115]–[120].

Los AGVs normalmente se mueven con libertad en las dos dimensiones y su alimentación energética es por medio de baterías. Las baterías son uno de los elementos clave en este tipo de robots y, gracias a los avances en materia de esta tecnología, en la actualidad los robots pueden tener mayor autonomía energética, lo cual les permite trabajar más horas de manera

continua, sumado a otro factor no menos importante, la posibilidad de que las baterías admiten ciclos de carga más rápidos sin que se vea resentida significativamente su vida útil.

Con todo, los AGVs parecen ser una tecnología punta, que desde el punto de vista de transporte, puede contribuir a conferir mayor flexibilidad y adaptabilidad a las instalaciones (gestionar productos con diferentes características, permitir un cambio ágil del *layout* de la planta), mayor capacidad de evolucionar (agregar/actualizar las máquinas y el sistema de transporte adaptarse a ello) y conferir mayor robustez, entendido como tolerancia a los fallos (dado que los AGVs pueden ser fácilmente reemplazados o quitados de la red de transporte en caso de fallo), entre muchas otras ventajas.

5.1.1 APLICACIÓN EN LABORATORIOS AUTOMATIZADOS

Los laboratorios de análisis clínicos (LAC) han sido progresivamente transformados en entornos de 'manufactura' o producción complejos, capaces de producir miles de análisis por hora, tanto de sangre como de otros fluidos biológicos. En este contexto, las muestras son puestas en tubos que son colocados en racks los cuales son llevados a diferentes máquinas encargadas de analizar las muestras, llamados analizadores. Normalmente el sistema de transporte más común es el de las cintas transportadoras [121].

Desafortunadamente, desde la asignación de la muestra, el transporte de la misma y la finalización de la orden asignada, se agregan otros factores que hacen incrementar la complejidad del sistema, dado que algunos *tests* tienen que ser repetidos, hay muestras que tienen prioridad de procesado, no todos los racks tienen que ser llevados al mismo analizador y varios analizadores pueden realizar el mismo trabajo pero con diferente capacidad de carga de trabajo.

Como resultado, la complejidad de gestionar esta clase de laboratorios es algo compleja, aún cuando utilizan una infraestructura para el transporte relativamente simple. La idea del prototipo presentado en esta sección es la de reemplazar las cintas transportadoras con pequeños AGVs, donde existen algunos casos de éxito que se pueden nombrar tales como en [122] y [123], entre muchos otros. Utilizar AGVs permite agregar un mayor grado de libertad al sistema de transporte y en consecuencia al sistema en general, ya que al poseer una cierta

autonomía pueden ayudar a disminuir la carga de trabajo en cuanto a la planificación en planta como también evitar que el sistema de gestión de la planta tenga que trabajar con gran cantidad de datos, contribuyendo todo ello a ganar en cuanto a flexibilidad, robustez ,etc. [124].

Para incluir la mayoría de las características de los laboratorios automatizados, el caso de estudio incluye 4 analizadores diferentes: uno para conteo de iones, otro para coagulometría y dos para análisis bioquímicos ya que la mayoría de las muestras a analizar requieren de este último.

En Fig. 12 se muestra un laboratorio automatizado de tecnología punta, donde el sistema de transporte (verde) utiliza cintas transportadoras.

Fig. 12 Sistema de transporte con cintas transportadoras de un LAC automatizado de tecnología punta [125].

El *layout* del sistema de transporte de la Fig. 12 es el que se plantea reemplazar (si es factible) por AGVs, tal cual se puede observar en el prototipo de la Fig. 14, simplificando la infraestructura utilizando un *layout*, donde las rutas están marcadas por líneas y donde existen puntos específicos de decisión denotado por marcas, bajo este contexto los AGVs se mueven por el *layout* siguiendo las líneas hasta que encuentran las marcas que le informan sobre qué acciones puede realizar en función también del objetivo asignado a cada robot.

5.1.2 VALIDACIÓN FUNCIONAL

En el caso de estudio de esta tesis validar funcionalmente implica chequear que la simulación ABM se comporta igual o de manera aproximada a la realidad y que será factible pasar al siguiente paso del proceso de diseño, el despliegue del prototipo del sistema.

Entre los aspectos a contemplar para la validación, se encuentran las características del sistema de transporte, tales como el número de AGVs, la velocidad media de los mismos, el tiempo de comunicaciones con el simulador ABM, su comportamiento ante uniones y bifurcaciones de la planta, etc. y, además, estudiar que todo el sistema simulado es capaz de cumplir con los requerimientos/especificaciones de la aplicación.

A continuación se muestra en la Fig. 13 la vista durante la simulación con el objetivo de validar la misma. Con el fin de hacer más descriptiva la figura y ofrecer más datos que son utilizados durante esta y posteriores secciones, se ha agregado información de la longitud de algunos segmentos (en cursiva) y la numeración de los nodos (valor cercano a la marca X). También se muestra el sentido de circulación de la red de transporte a través de flechas y algunos de los nodos que son considerados uniones, bifurcaciones y unión/bifurcación. Se pueden observar otros datos relacionados con parámetros de configuración de la simulación y la descripción de estados a medida que se está ejecutando.

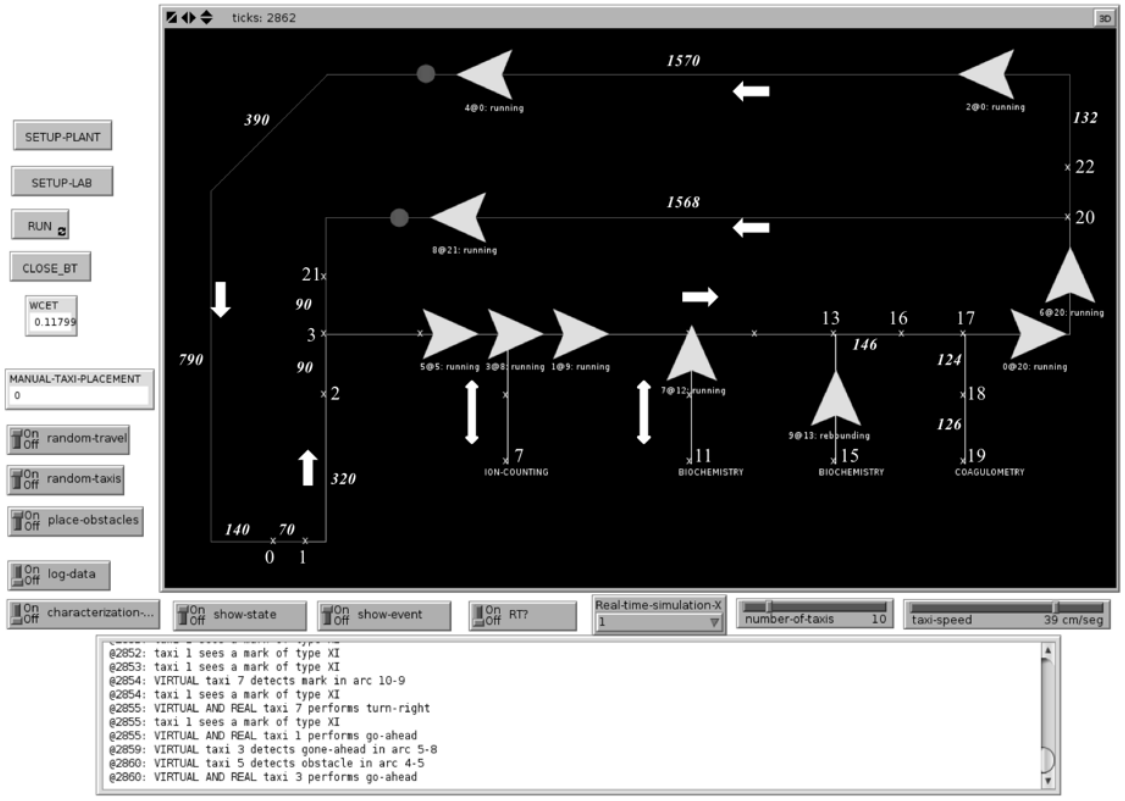


Fig. 13 Simulación de la planta durante validación

5.1.2.1 Estimación de las características del controlador

La simulación ABM correspondiente al modelado de la planta del laboratorio de análisis clínicos se ejecuta en un controlador, básicamente un ordenador. Para saber si el lazo de control principal (MCL) de la simulación es capaz de ejecutarse lo suficientemente rápido como para permitir que *n* AGVs puedan ser monitorizados y controlados desde el entorno simulado, es necesario medir el tiempo de ejecución de este MCL pero principalmente determinar dinámicamente (dada la complejidad del sistema) un valor que es considerado el peor caso en cuanto al tiempo de ejecución del MCL, denominado WCET. Esto es conveniente realizar como un primer paso antes del despliegue del sistema, dado que si bajo simulación el WCET es muy malo, es lógico que vaya a peor al intentar que la simulación interactúe con la realidad.

Para estudiar la máxima carga que es capaz de soportar el controlador funcionando en tiempo real, se realizaron una serie de simulaciones con 20 AGVs realizando órdenes de transporte

aleatorias, con el objetivo de determinar el WCET (peor tiempo de ejecución del MCL). De estos ensayos se reportó un WCET del sistema de 16ms.

Por otro lado, se analizó el tiempo relacionado con las comunicaciones por Bluetooth® con 4 transceptores (1 por AGV) y se proyectó una línea de tendencia la cual reportó un máximo de 20 AGVs para un tiempo de comunicaciones de alrededor de 20ms, dicho valor se suma a los 16ms. de simulación, lo que reporta un total de 36ms.

Con todo, el MCL es capaz de ejecutarse a una frecuencia de 28Hz. Por lo tanto, el controlador del simulador ABM podría realizar SCADA en tiempo real a frecuencias no mayores a 14Hz. Dado que el controlador debe ser por lo menos dos veces más rápido que el componente más rápido de la realidad a supervisar y controlar, para evitar pérdida de datos de monitorización y poder enviar comandos de control que sean captados dentro del mismo bucle del controlador del componente a controlar.

Con ello, a una frecuencia de 14Hz, se puede tener una resolución espacial de alrededor de 7mm. para un robot que navega a unos 10cm/s.

5.1.2.2 Validación de los agentes de transporte

Los agentes de transporte están desarrollados e implementados mediante el enfoque ALOHA. Para validarlos se siguen dos etapas: la primera, relacionada con su validación a nivel de simulación y la segunda cuando el agente está completo, lo que implica que ya le ha sido integrado el componente de la realidad.

La validación a nivel de simulación requiere que el modelo virtual (L_V) sea representativo de su contraparte de la realidad (L_R). Para ello, uno de los aspectos a contemplar son las características funcionales y no funcionales del mismo. En cuanto a las primeras, están relacionadas con datos estáticos de la red de tráfico y las características de los AGVs tales como velocidad media, consumo de energía, etc., las cuales podrían ser introducidas en el modelo como valores iniciales a partir de datos nominales (especificaciones técnicas de los componentes de la realidad). Las segundas se relacionan más bien con criterios que se pueden

usar para juzgar la operación del sistema, en este caso sería, por ejemplo, la caracterización de los robots y cómo afectan a la calidad de la sincronización.

Para la validación funcional en simulación ha sido suficiente con implementar los modelos de los AGVs con parámetros nominales y dado que se asume que la red de tráfico es constante y que está definida topológicamente por un grafo que es conocido por todos los agentes de transporte, durante la etapa de validación funcional de los mismos, estos se comportaron según lo esperado.

Sin embargo, para la validación no funcional, que está más relacionada con el control de la planta real, se tuvieron que realizar una serie de ensayos [126] para medir parámetros de manera más precisa, ya que los mismos tienen influencia en la calidad de los aspectos no funcionales. Dado que la precisión del modelo depende de una buena caracterización de la planta y cuanto más precisa sea esta, la calidad de la sincronización entre el modelo simulado y la realidad será mayor. Una sincronización de buena calidad es aquella que evita retardos innecesarios en la simulación o interacciones extras en el lazo de simulación que le permita alcanzar a la realidad.

De manera simple, al caracterizar lo que se hace es medir el tiempo que tarda el agente de transporte en navegar de un nodo/marca a otro y el tiempo utilizado sobre el nodo mientras decide cuál es su siguiente destino.

Para ello, la caracterización comienza mediante una solicitud desde el alto nivel del agente (*H*) hacia el L_R donde se mide el tiempo que tarda en responder L_R al *H*. Esto se realiza para cada segmento de la planta y el valor obtenido es comparado con el registro previo del tiempo de ese nodo para que según ciertos criterios se decida mantener el valor registrado o actualizarlo con este nuevo valor, todo ello con el objetivo de hacer que el modelo en la simulación L_V tenga un comportamiento concurrente/sincronizado con la realidad.

5.1.2.3 Requisitos de la aplicación y análisis de validación

Los analizadores de tecnología punta para laboratorios de análisis clínicos automatizados, tal como el Cobas 8000 [127] tiene una producción/rendimiento medio de 12000 muestras por

día y, dado que los AGVs son capaces de transportar 1 rack con 5 muestras cada uno, esto implicaría que cada orden estaría finalizada al cabo de 36 segundos.

El tiempo de respuesta para procesar una orden es función de varios parámetros, tales como las características de los AGVs (velocidad, capacidad de carga, etc.), la topología de la planta, el rendimiento de los analizadores, y las distancias que tienen que recorrer los AGVs (por ejemplo, si tienen que ir a cada analizador, usar la línea de recirculación, etc.).

Para un cálculo realista se tendrá en cuenta que nuestra plataforma (Fig. 14) es dos veces más pequeña de lo que sería la planta real. Es por ello que todos los valores de distancias y en consecuencia de tiempo se encuentran multiplicados por dos.

En el peor caso, para completar una orden el AGV tiene que recorrer una distancia de 22 metros (11 x 2), esta circunstancia ocurre cuando luego de recibir la orden el AGV tiene que ingresar a cada analizador, al finalizar cada uno de ellos deben utilizar la línea de recirculación para luego pasar por los puntos (5, 9, 13, 17) sin ingresar a los analizadores dirigiéndose directamente hacia el punto de partida (0). En este caso el tiempo consumido sería de 240,8 segundos (120,4 x 2) más 36 segundos que es el tiempo empleado por cada analizador para procesar una orden, por lo tanto, considerando estos factores, el AGV utiliza 276,8s. para completar la orden.

En el mejor de los casos, el AGV tiene que recorrer una distancia de 14,6 metros, que sería el caso en que ingresa únicamente a un analizador y no utiliza la línea de recirculación, retornando al punto de partida (0), de esta manera, el tiempo total consumido (navegación del AGV + tiempo de procesado del analizador) es de unos 195,8s. Por lo tanto, el tiempo promedio de una orden es de alrededor de 236,3 segundos.

La carga típica de trabajo de laboratorios de análisis clínicos automatizados instalados en hospitales con tecnología punta es alrededor de 1 millón muestras o 10 millones de tests/determinaciones al año [121], lo que da un rendimiento medio de 4167 muestras/día considerando únicamente los días laborables.

El rendimiento productivo de estos laboratorios depende por lo menos de los siguientes parámetros, la topología de la planta, el tiempo de respuesta de una orden, el número de

analizadores disponibles y su rendimiento medio, el número de agentes de transporte y sus características técnicas, la coordinación entre ellos, la inteligencia del LIMS al momento de asignar las órdenes a los AGVs, la manera en que el LIMS organiza las muestras y la combinación de todos estos parámetros, entre otros.

Dada la complejidad para calcular el rendimiento, se proponen algunos escenarios simples para intentar pronosticar si el sistema propuesto con el *framework* de esta tesis es capaz de gestionar la carga de trabajo de los laboratorios de análisis clínicos actuales.

Suponiendo que cada AGV tiene una autonomía de las baterías de 8hs. y que el tiempo empleado para recargarlas es el mismo, para cubrir un día de trabajo se necesitan al menos dos AGVs, uno que se mueve por la planta y el otro recargando las baterías y alternándose entre estas situaciones entre ellos. Este escenario nos provee 1 AGV llevando muestras las 24hs/día.

Bajo este escenario, y tomando el peor tiempo de respuesta de una orden, se obtiene que el rendimiento más bajo es de 1560 muestras/día, lo cual cubre alrededor del 37,4% de la carga diaria de trabajo (alrededor de 4167 muestras/día).

En el caso que 3 AGVs transporten órdenes las 24hs/día y que la separación entre ellos sea de por lo menos 4 metros (para evitar posibles retardos relacionados con la coordinación entre los agentes de transporte y los analizadores) el rendimiento esperado del sistema es de 4680 muestras/día, lo cual excede en un 12,3% la carga de trabajo esperada.

Bajo este escenario, se necesitarían 6 AGVs funcionando simultáneamente en tiempo real, lo cual es factible dado que el sistema es capaz de soportar hasta 20 AGVs bajo estas condiciones.

Con todo, se puede validar funcionalmente el *framework* de esta tesis ya que es capaz de gestionar en tiempo real la carga de trabajo actual de laboratorios de análisis clínicos de tecnología punta.

Por otro lado, dentro de la validación es importante analizar la formación de posibles cuellos de botella cuando varios AGVs se encuentran circulando ya que esto podría llevar a restricciones emergentes durante la ejecución del sistema. Uno de los factores que podrían contribuir a ello es la restricción en la movilidad de los robots dada la característica

unidimensional del *layout* de la planta sumado a ciertos puntos, como los que representan uniones (por ejemplo el punto 3).

Dado que por restricciones de tiempo real la plataforma la máxima cantidad de robots simultáneamente ejecutándose es de 20, lo cual representa el escenario más crítico en cuanto a congestionamiento de tráfico, es este el escenario elegido para el estudio de los cuellos de botella. Para ello se distribuyen 15 AGVs en la línea más hacia arriba del *layout* y 5 AGVs en la línea más hacia la izquierda del *layout*. Bajo estas circunstancias se probarán dos escenarios, el primero donde cada AGV tiene que cumplir con órdenes aleatorias y el segundo donde cada AGV tiene órdenes que le fuerzan a ingresar a cada uno de los analizadores. Bajo ambos escenarios el cuello de botella emergente se da en el nodo/marca (3) y a veces en marcas donde el AGV intenta ingresar en un analizador que está todavía en uso. Por consiguiente, luego de tres simulaciones en no tiempo real, lo que significa que el tiempo era medido en *ticks*, hasta llegar a los 9050 cada simulación, los resultados muestran una concentración de AGVs alrededor del área del nodo 3 de un 15% mayor para el escenario con órdenes aleatorias, en comparación con el escenario sin órdenes de este tipo.

Cabe destacar que estos ensayos solo muestran la existencia de estos cuellos de botella y donde es más factible encontrarlos, pero otros resultados se podrían encontrar para diferentes valores de *ticks*, diferentes posiciones iniciales de los AGVs, diferentes velocidades de circulación de cada uno, etc.

Finalmente, en caso que estos cuellos de botella constituyan una limitación crítica, los mismos podrían ser evitados modificando ligeramente el *layout* de la planta (por ejemplo con una nueva línea de recirculación), dotando a los AGVs de sensores que le permitan moverse en 2D, eliminando las líneas del *layout* y convirtiéndolo en un entorno de 2D, mediante una mejor planificación de las órdenes a asignar, etc.

5.1.3 DESPLIEGUE DEL SISTEMA

Para que el sistema quede completamente desplegado todos los niveles asociados al enfoque ALOHA tienen que ser completados. En este punto, únicamente queda trasladar/reprogramar/empotrar el código de bajo nivel correspondiente a la representación

virtual del AGV (L_{Vi}) en el AGV real (L_{Ri}), realizando esto para lo *i* AGVs que integran la solución final.

Para ello, dado que el prototipo ha sido implementado con Parallax Boebots representando los AGVs, el código del L_V debe ser portado a PBASIC (L_R) para que pueda ser interpretado por el controlador *Basic Stamp* de los robots.

Por lo tanto, el prototipo del sistema se encuentra listo tan pronto como el agente ALOHA está completo y algún entorno con el cual interactuar también ha sido creado para ellos

El prototipo resultante se puede ver en la Fig. 14 compuesto por los robots con el software embebido, una red de tráfico impresa en la superficie marcando las rutas de movimiento de los AGVs, un ordenador con Netlogo ejecutando el simulador ABM como SCADA en tiempo real y unos dispositivos de comunicación inalámbrica compuesto por módulos Bluetooth®, además algunos videos relacionados con su funcionalidad se pueden ver en [126], [128], [129].

La posición de los robots en la planta está determinada por marcas en el *layout* que representan la entrada a un grafo dirigido que le dice a los robots cómo gestionar el siguiente movimiento de acuerdo a la marca en la que se encuentran y las características de la orden de trabajo que tienen que completar. Este tipo de odometría permite a los robots conocer su posición entre marcas, pero no su posición exacta entre ellas (para nuestro objetivo no era necesario). Los robots para navegar utilizan dos tipos de sensores, el primero es un conjunto de sensores que le permite seguir las líneas y detectar las marcas, el segundo es un sensor tipo sonar para la detección de obstáculos.

El movimiento de los robots en la planta comienza con la asignación de una nueva orden y la carga de un rack con 5 muestras a analizar en la marca (0), luego el robot se mueve autónomamente para cumplir la orden hasta llegar a la marca (20), que es donde hay una comunicación entre el LIMS y el AGV y, según que *tests* han sido realizados y cuáles faltan hacer, el AGV continúa hacia la línea de retorno (22) o bien hacia la línea de recirculación entre las marcas (20) y (3). En este segmento es donde el AGV espera su turno por diversos motivos, por ejemplo, el analizador estaba ocupado para el test que necesitaba realizar y tuvo que ir a esta zona de espera, hubo un error al procesar la muestra y tiene que esperar su turno para que sea re-procesada, etc.

Finalmente, al comienzo de la línea de retorno en la marca (22), los AGVs ya han descargado los racks y se dirigen hacia la cola de espera para recibir nuevas órdenes con las baterías ya recargadas, si ello fuese necesario.

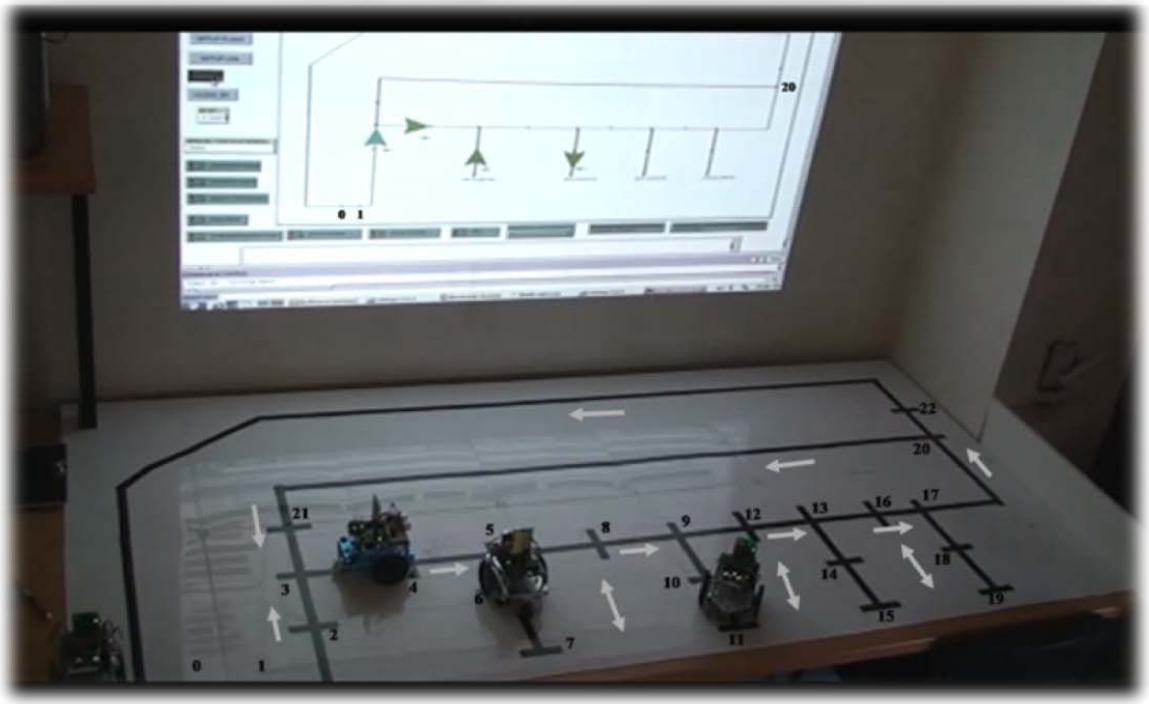


Fig. 14 Despliegue del prototipo en realidad mixta

En la Fig. 14 se puede apreciar en la parte inferior el prototipo de la plataforma con 4 analizadores diferentes (marcas 7, 11, 15 y 19), también se puede apreciar el sentido de circulación de la red de tráfico así como puntos de bifurcación (20), unión (3) y bifurcación/unión (5, 9, 13, 17). En la parte superior se encuentra el simulador ABM utilizado para SCADA de la planta real, donde se pueden ver 4 AGVs, ya que en este ejemplo la plataforma está funcionando en realidad mixta, lo que significa que no todos los AGVs virtuales tienen su contraparte real. Esto es de especial utilidad cuando se quieren estudiar comportamientos con un mayor número de robots de los que realmente se posee físicamente.

5.1.4 ESTUDIO DEL SISTEMA DESPLEGADO

Para validar funcionalmente la plataforma, existen dos factores principales a analizar, uno de ellos está relacionado con la capacidad que el simulador ABM pueda realizar la supervisión el control y la adquisición de datos en tiempo real y el otro factor relacionado con la caracterización de los agentes de transporte, el cual tiene una interdependencia con el sincronizador del mundo virtual y el real. Es por ello los citados factores se estudiarán en los siguientes apartados:

5.1.4.1 Caracterización off-line de los AGVs.

El proceso de caracterización consiste en medir y luego analizar el tiempo de navegación entre marcas para cada uno de los robots móviles (AGVs) a lo largo del circuito mostrado en la Fig. 14. Para ello se realizaron 30 ejecuciones y se obtuvieron los datos para cada robot en cada segmento como también el promedio del tiempo total de todo el recorrido. El circuito completo posee una longitud de 11 metros y el tiempo promedio para cada robot para completar el circuito ha sido de 112,654; 125,456; 123,169 segundos correspondiente a los robots llamados I, F y L respectivamente.

A pesar que el *framework* permite una actualización online de estos datos, es importante realizar esta caracterización previa por robot para poder obtener una simulación mejor sincronizada con la realidad desde el inicio de la ejecución online.

Además, esta caracterización off-line constituye la primera oportunidad para detectar la influencia de ruidos de diferentes fuentes tales como aquellas relacionadas con la comunicación inalámbrica y la de los sensores tanto de detección de línea como de obstáculos. En el primer caso, no se detectaron fallos durante la ejecución relacionados con los transceptores Bluetooth® tanto de los AGVs como del ordenador, esto se puede deber a que están suficientemente cerca uno de otros como para mantener la relación señal-ruido dentro de valores que no causen ningún fallo, en la misma línea, tampoco se detectaron interferencias entre los transceptores de los AGVs y la influencia de otras fuentes electromagnéticas. En el segundo caso, se detectó influencia de ruido en los sensores de línea, debido a fuentes tales

como luz fluorescente y solar, lo cual fue solucionado protegiendo los sensores de dicha influencia.

En la Fig. 15 se muestran los datos de caracterización correspondientes a cada robot para cada segmento de la planta. En el eje vertical se muestran los segundos para cada segmento, en el horizontal el segmento correspondiente y en profundidad el número de ensayo correspondiente para cada robot.

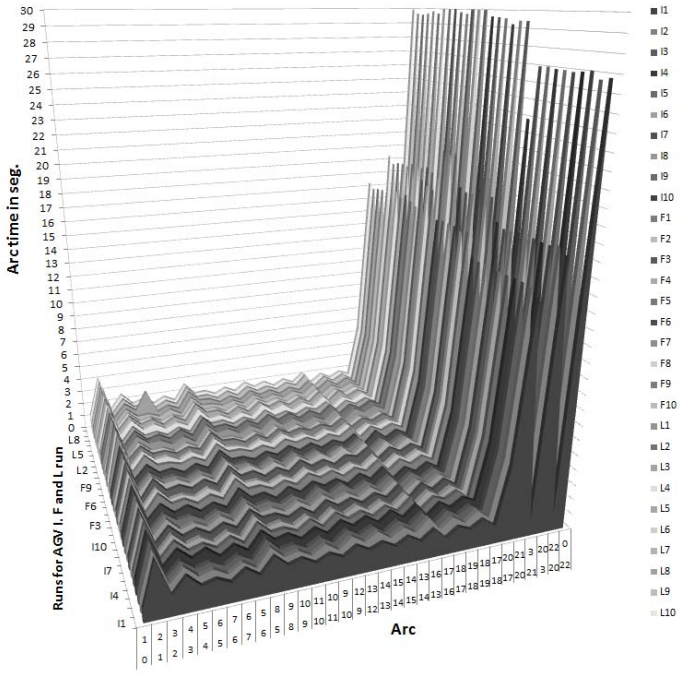


Fig. 15 Caracterización de cada robot para cada segmento de la planta

En la misma se puede observar un comportamiento más bien homogéneo para cada robot a lo largo de los recorridos que tuvieron que hacer, a pesar que a medida que los recorridos aumentaban la autonomía de las baterías iba decreciendo.

5.1.4.2 Estudio del WCET del sistema

Durante el proceso de caracterización del Sistema, se ha aprovechado para medir y validar el WCET del sistema. Este estudio se realiza dado que el WCET se encuentra relacionado con el

bucle de control principal, que debe ser ejecutado lo suficientemente rápido, para permitir realizar SCADA utilizando el simulador ABM sin pérdida de información.

El WCET se ve influenciado por la velocidad de ejecución del controlador, la cantidad de componentes de la realidad a supervisar y controlar, la manera de representar la realidad en la simulación, el modelo de cómputo del componente virtualizado, las características del sistema operativo donde se ejecuta el programa, las prioridades de ejecución de los servicios lanzados por el sistema, entre otros.

Como se puede apreciar en la Fig. 16 el WCET del sistema es de 500ms. lo cual no satisface para SCADA en tiempo real de múltiples robots, dado que los mismos se mueven a una velocidad de unos 10 cm/s y con ello no se puede obtener un monitoreo y control para la menor distancia posible a detectar correspondiente a los 18mm. del ancho de las marcas. No obstante, este problema puede ser evitado utilizando un ordenador más potente, un sistema operativo en tiempo real o con la menor cantidad de servicios ejecutándose o que puedan ser lanzados y que interfieran con la ejecución del programa para SCADA. Sin embargo, hay que tener en cuenta 1) que el sincronizador podría hacer que el robot real espere a la simulación y, 2) que el WCET y otros valores de ejecución del lazo de control principal por arriba de 50ms. representan únicamente el 1,3% del total de los ciclos de control, con lo cual parece no ser un gran inconveniente ejecutar el simulador ABM para SCADA con la infraestructura de este *framework*.

A pesar que en la Fig. 16 se presta especial atención en el WCET, se puede observar también la distribución del tiempo de ejecución del bucle principal para cada robot, donde se puede ver que en un gran porcentaje este tiempo se encuentra por debajo de los 150ms.

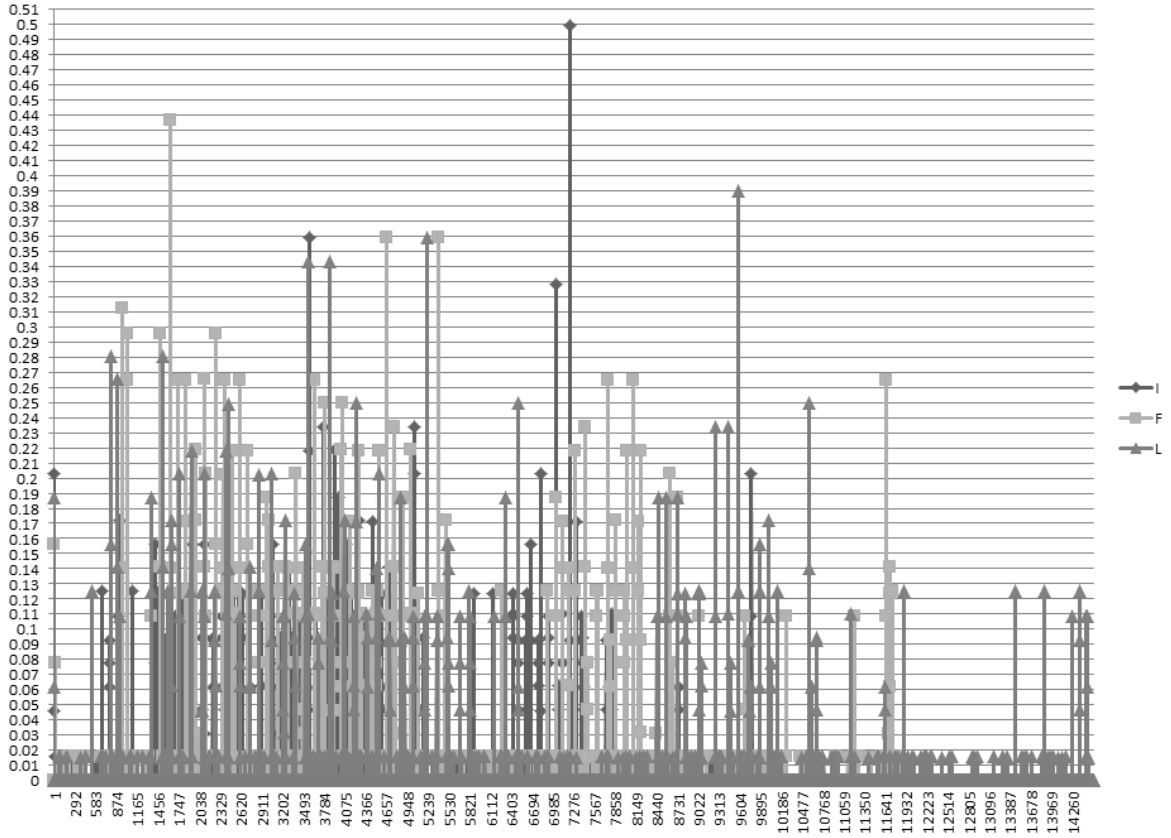


Fig. 16 WCET para tres AGVs durante la caracterización

En la Fig. 16 se muestra en el eje de las ordenadas el tiempo de ejecución del bucle principal representado en segundos, mientras que en el eje de las abscisas se observa la evolución del contador de lazos de control principal mientras se realiza SCADA en tiempo real de la planta; los datos son mostrado para los tres robots caracterizados (I, F y L).

Para garantizar el monitoreo y control en tiempo real, todos los tiempos de ejecución del lazo de control principal son comparados con el WCET admitido por el sistema, siendo este valor aquel que aún siendo el peor caso, todavía permite realizar SCADA en tiempo real sin pérdida de información. Por lo tanto, el lazo de control tiene que tener un período compatible con robots cuyos controladores embebidos sean capaces de comprender instrucciones algo complejas y donde el tiempo de ejecución de esas instrucciones sea mayor que el WCET del modelo.

En caso que el tiempo de ejecución de los robots sea cercano al WCET, existen alternativas para preservar coherencia entre la simulación y la realidad, tales como incluir marcas de tiempo en los mensajes o bien minimizar el WCET modificando apropiadamente el *scheduling* de la ejecución del agente [53].

5.1.4.3 Calidad de la sincronización

Cuanto más precisa es la caracterización de los componentes físicos de la planta, mejor será el control y consecuentemente, la eficiencia del sistema.

La calidad de la sincronización a lo largo de la caracterización *off-line* es medida en términos de porcentaje que el sincronizador ha estado en cada uno de los estados, respecto al número total de estados de HSE que acontecen. Como es de esperar, la simulación tiende a ser optimista y va por delante de la realidad alrededor del 33% del tiempo, influenciado porque el retardo en los segmentos es inicialmente cero. Sin embargo, la mayor parte del tiempo (66%), la simulación se ejecuta en concurrencia con la realidad y únicamente un 1% del tiempo se ejecutan ISM para hacer que la simulación alcance a la realidad.

En la Fig. 17 se puede apreciar la calidad de la sincronización para cada uno de los robots en cada uno de los 10 ensayos de cada robot, así como un promedio de las 10 ejecuciones para cada robot.

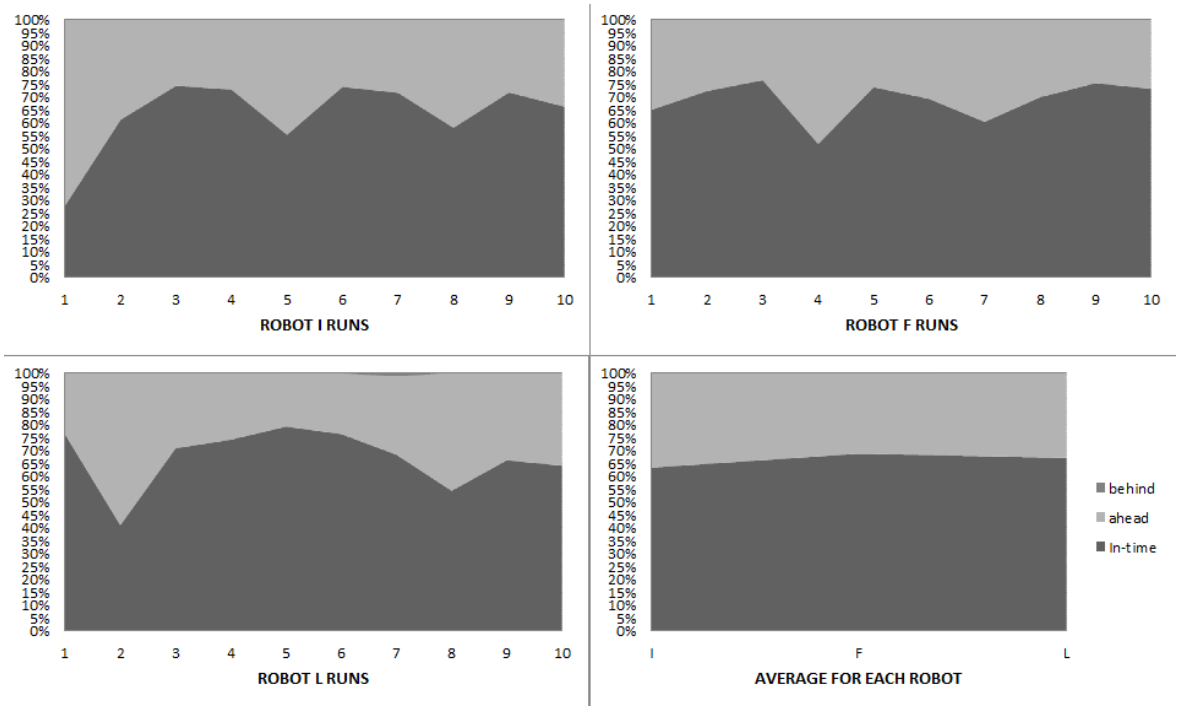


Fig. 17 Distribución de la calidad de sincronización por robot y en promedio

Cabe destacar que a pesar de que el porcentaje *in-time* no es muy alto (66%), un gran porcentaje del estado *ahead* de la simulación respecto a la realidad puede trasladarse al estado in-time mejorando la calidad de la caracterización, por ejemplo haciendo que esta se actualice por medio de diferentes estrategias a medida que se ejecuta la plataforma. Es necesario contemplar que el estado menos deseado es el *behind*, ya que aquí la simulación va por detrás de la realidad, causando a la realidad una espera innecesaria hasta que se ejecuta mediante simulación la estrategia de ISM que hace avanzar rápidamente la simulación hasta el punto del evento donde se encuentra la realidad.

5.1.5 CONCLUSIONES

En este caso de estudio se ha mostrado que es posible realizar SCADA en tiempo real en aplicaciones del dominio industrial para un caso de sistema de transporte con AGVs utilizando un simulador ABM mediante el enfoque ALOHA desarrollado a lo largo de esta tesis.

Para ello se siguió la metodología sugerida para el despliegue de sistemas basados en ALOHA, donde se validó el caso a nivel funcional y luego durante la etapa de despliegue se ha embebido el modelo de bajo nivel (L_V) en el componente real (L_R), en este caso los AGVs.

Se probó la plataforma tanto en sincronización de la simulación con la realidad así como bajo situaciones de realidad mixta mediante un conjunto de varios robots y se estudió la factibilidad de poder extender su aplicabilidad hasta 20 AGVs en tiempo real.

Con todo, se muestra que es factible y conveniente utilizar el enfoque ALOHA ya que permite agilizar el proceso que va desde el diseño hasta el despliegue del sistema, dado que, el mismo código verificado y validado en simulación es el que se utiliza para el control y la supervisión de la planta real, realizando únicamente una transformación del código de bajo nivel (L_V) del agente en su contraparte de la realidad (L_R).

En la siguiente sección se muestra un caso de estudio que consiste en un sistema de producción robotizado, el cual se utiliza como un segundo demostrador y que valida el *framework* desarrollado en esta tesis a la vez que ha permitido tener una aproximación sobre el tiempo de desarrollo empleando las herramientas de esta tesis.

5.2 Sistema de producción robotizado

Desde la revolución industrial hasta la actualidad, la automatización del área industrial ha ido creciendo, se podría decir, exponencialmente. La robótica ha tenido, tiene y tendrá un gran papel en esta área. El uso de robots para tareas de producción, permite que las industrias puedan producir bienes de manera continua y con mínima intervención humana. Sin embargo, los requerimientos actuales del mercado hacen que las industrias tengan que contemplar ciertas características como adaptabilidad, robustez, flexibilidad, etc., para poder mantenerse competitivas y perpetuar su existencia en un entorno donde el cambio es moneda constante.

Por ejemplo, tienen que ser capaces de producir diferentes productos, y por ende adaptar sus recursos de manera ágil a ello. Por otra parte, la demanda de productos se ve muchas veces incrementada y por ende necesitan agregar más recursos del tipo *plug and produce* para atender estos incrementos. Al mismo tiempo, recursos complejos como lo son los robots, tienden a ser cada vez más autónomos, sin embargo necesitan estar comunicados con otros recursos (robots, otros dispositivos, sistemas de planificación y asignación de tareas, etc.) para poder llevar a cabo las tareas relacionadas con el proceso productivo y los objetivos de producción. Con todo, el sistema se vuelve más complejo, dado que los nodos que lo integran a pesar de su autonomía creciente tienen que estar interconectados, funcionar de manera armónica y robusta (libre de fallos), entre otras cosas. Por otra parte, es común que los sistemas industriales tengan salas especializadas para poder supervisar a nivel global qué es lo que está sucediendo en la planta, donde a veces, los mismos sistemas no solo son utilizados para la supervisión, sino también para el control y la adquisición de datos (SCADA). A continuación se presenta el prototipo utilizado para SCADA implementado con el enfoque ALOHA y que sirve para validar su aplicabilidad.

5.2.1 DESCRIPCIÓN DE LA PLATAFORMA

La Fig. 18 muestra un esquema donde se describe el prototipo, el cual está integrado por dos ordenadores, el primero donde se ejecuta el simulador basado en ABM utilizando Netlogo, el cual actúa como HMI y se utiliza como SCADA de la planta real. La función del segundo ordenador es el de hacer de proxy permitiendo la comunicación entre el simulador y la planta real. El proxy se encarga de traducir los mensajes en ambos sentidos, lo que significa entre el sistema SCADA y el sistema embebido que hace de interfaz con sensores (por ejemplo finales de carrera) y los actuadores neumáticos del brazo robot de la celda de producción (plataforma). La secuencia de producción comienza desde el lado izquierdo (color negro) y mueve una pequeña pelota hacia el lado derecho (color gris claro), para luego retornar hacia el punto inicial en busca de una nueva pelotita. Para mostrar la dirección del movimiento, se muestra una posición intermedia del brazo robot en color gris claro. En la figura también se pueden ver las dimensiones de la plataforma cuya especificación en cuanto al actuador neumático del brazo robot corresponde a una presión de 10 bars.

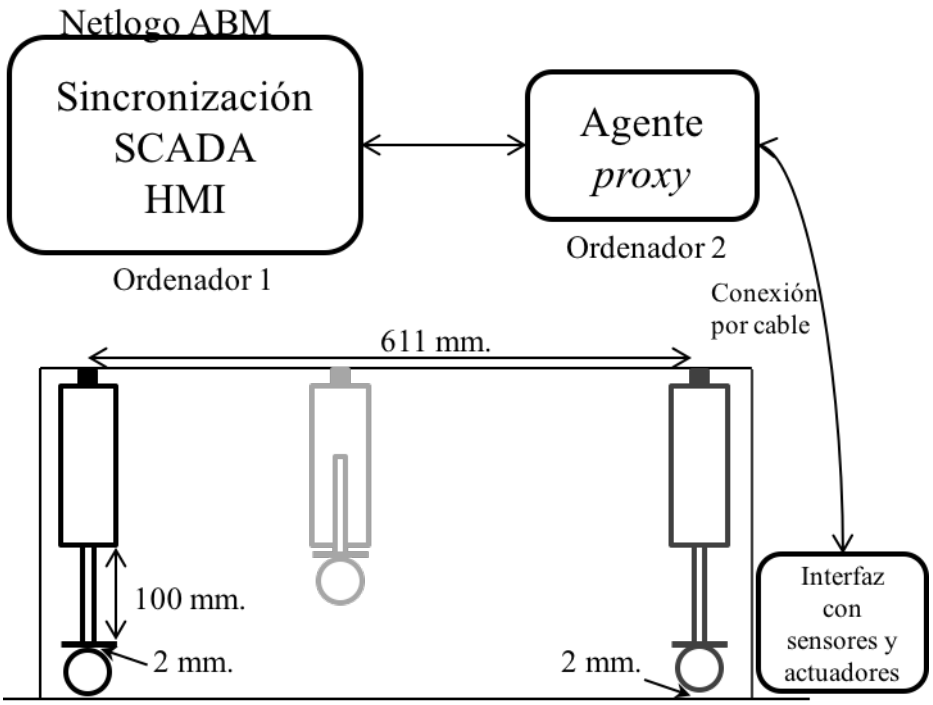


Fig. 18 Esquema de la plataforma prototipo

Cuando la planta se encuentra en producción, el brazo robot recibe comandos de alto nivel y luego realiza una serie de secuencias según el tiempo de comando. Las secuencias a seguir por el brazo están asociadas con la tarea asignada y ésta última es especificada a través del comando de alto nivel enviado al brazo, la secuencia también se ve influenciada por las condiciones iniciales. Por lo tanto, la tarea/meta para ser lograda por el agente robot posee un universo dinámico de soluciones. Un ejemplo de lo que sería una tarea es descripto a continuación.

En los ensayos (compuestos por varias tareas), la meta del agente robot es mover un objeto de la izquierda hacia la derecha y luego volver por el siguiente objeto. El estado inicial del brazo robot es hacia la izquierda y hacia abajo y con el elemento de succión del objeto en *off* (0). Comenzando desde este estado inicial (0), las siguientes secuencias son: 1) se acciona el elemento de succión (SO), 2) se sube el brazo (UA), 3) mover el brazo hacia la derecha (RA), 4) bajar el brazo (DA), 5) poner en *off* el elemento de succión (SF), 6) mover el brazo hacia la izquierda (LA), y por ende el brazo ahora se encuentra nuevamente en la posición inicial (0).

Por lo tanto, una tarea específica está compuesta por esta serie de secuencias. El total de distancia recorrida en cada tarea es la suma de cada secuencia la cual arriba a 1426 milímetros.

A continuación, y antes de realizar el despliegue del sistema, se presentan algunos resultados de simulación.

5.2.2 RESULTADOS DE SIMULACIÓN

En esta sección, se presentan los resultados de simulación implementado mediante ABM, correspondiente a la Fig. 18 (esquema de la plataforma) y Fig. 21 (despliegue de la plataforma) y el análisis de los mismos.

El enfoque del estudio de simulación se centra en cómo varía el tiempo de ejecución del lazo de supervisión y control principal como función del número de agentes que integran la simulación. Esto puede proveer un pronóstico sobre la factibilidad de realizar SCADA de la realidad con más de un agente.

Las simulaciones han sido realizadas en el mismo ordenador utilizado para SCADA de la planta real y son ejecutadas 45 tareas en cada ensayo. Los ensayos consisten en contar el número de bucles del lazo de control principal (MCL) para diferentes brechas de tiempo así como el total de MCL, a medida que se van agregando agentes desde 1 hasta 5. La simulación se ha realizado en el peor caso desde el punto de vista del tiempo del MCL, pero para el mejor caso en cuanto a representación en HMI. Este escenario sucede cuando la simulación ABM es ejecutada en 3D en vez de 2D, dado que la representación en 3D es más demandante de cómputo.

La Tabla 1 Número de MCL como función del número de agentes durante 45 tareas realizadas por cada uno muestras el número MCL total y para diferentes ventanas de tiempo para diferentes números de agentes simultáneamente ejecutándose durante 45 tareas.

Agents	Loops total	Loops =< 10ms	10ms < Loops <= 50ms	Loops > 50ms
1	8112480	8087964	24496	20
2	4307818	4281191	26611	16
3	2667104	2640209	26885	10
4	1844009	1816720	27339	30
5	1327691	1300946	26694	51

Tabla 1 Número de MCL como función del número de agentes durante 45 tareas realizadas por cada uno

Como se puede observar en la Tabla 1 Número de MCL como función del número de agentes durante 45 tareas realizadas por cada uno, el número de lazos de control decrece a medida que el número de agentes es incrementado, esto podría estar asociado a que cada MCL consume un mayor tiempo durante su ejecución.

Una tarea consume 11,743 segundos para ser completada en tiempo real. Por lo tanto, este número debería ser cumplido por cada agente para que su representación en la HMI sea concurrente con la realidad.

La Fig. 19 ha sido creada utilizando datos de MCL menores o iguales a 10ms. (representando alrededor del 99% del total de MCL), el tiempo consumido por una tarea (11,743 segundos) y el número de agentes en cada escenario. El fin de la Fig. 19 es el de mostrar cómo varía el tiempo de cada MCL como al ir incrementando el número de agentes.

En línea sólida se muestras el promedio de tiempo de un MCL de simulación entre 1 y 5 agentes. En línea de puntos, se presenta una tendencia hasta 20 agentes mostrando el límite de agregado de agentes estando limitado por el hecho que cada MCL se vuelve lo suficientemente lento como para permitir SCADA para el caso más crítico de distancia (2mm. relacionado el *on/off* de la ventosa del brazo robot).

Como se puede apreciar en la Fig. 19, una ecuación cuadrática representa el comportamiento del tiempo consumido por cada MCL como función de la cantidad de agentes simulados simultáneamente ejecutándose.

Como resultado, cuando 20 agentes se ejecutan de manera concurrente, cada MCL consume alrededor de 4,25ms., lo cual se encuentra en el límite de lo aceptable para las especificaciones del brazo robot.

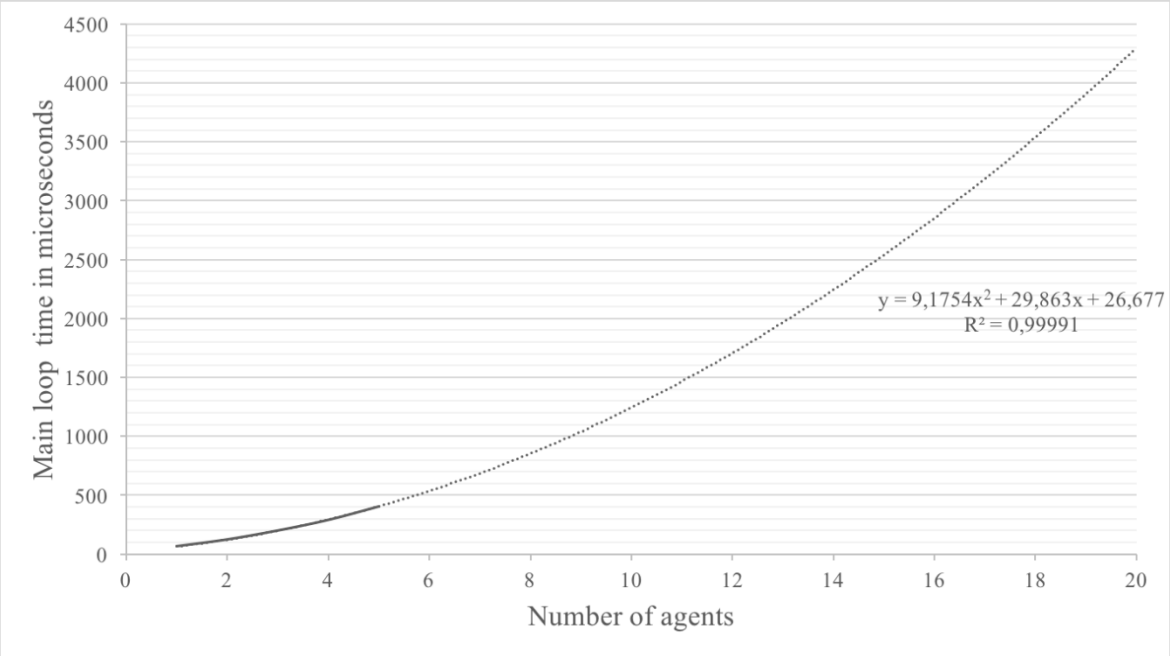


Fig. 19 Tiempo promedio del MCL como función de agregar agentes.

Como se muestra en la Fig. 19, de lo que es la simulación y la línea de tendencia, se espera que alrededor de 20 agentes puedan cumplir con las restricciones de tiempo real. Sin embargo, cuando la simulación se ejecuta concurrentemente con la realidad realizando SCADA de la misma, existen otros parámetros de influencia en las restricciones de tiempo real y que provienen de la realidad, como ser, la estructura del mensaje/protocolo de comunicación, la velocidad de cómputo de los controladores embebidos, etc. [130]–[133], lo cual puede hacer disminuir el número soportado de agentes tipo ALOHA soportados por el sistema.

Finalmente, en la Fig. 20 se muestra una representación 3D del escenario de simulación mientras 5 agentes se ejecutan simultáneamente.

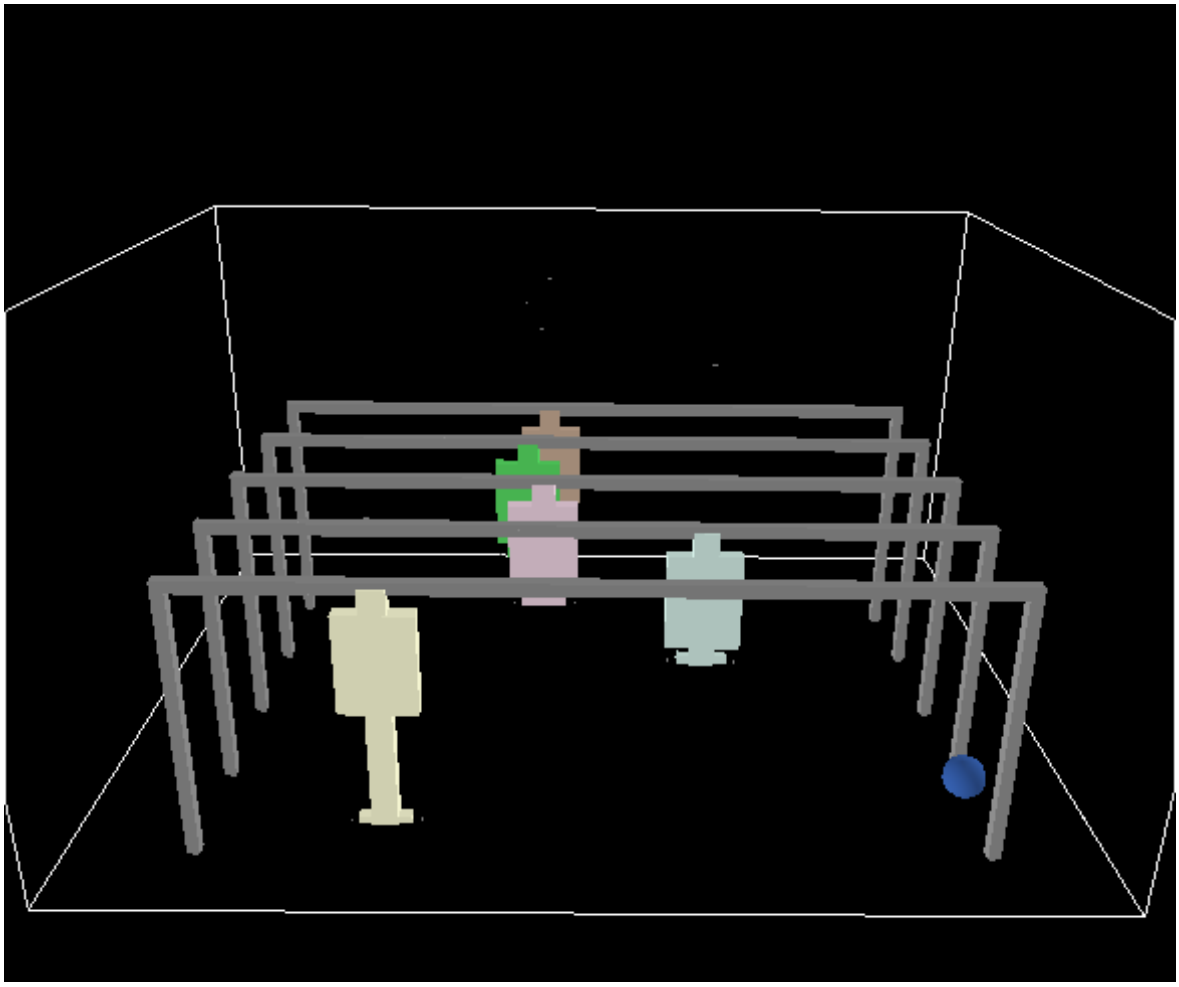


Fig. 20 HMI en 3D para simulación de 5 agentes ejecutándose simultáneamente

Cabe destacar, que más de 20 agentes podrían ejecutarse simultáneamente en el caso que la representación sea en 2D, dado que experimentalmente se ha determinado que la representación en 2D es alrededor de un 50% más rápida que la representación en 3D.

5.2.3 DESPLIEGUE DEL SISTEMA

El esquema de la planta mostrado en la Fig. 18 puede ser visto como una celda de producción que usa un brazo robot neumático, el cual succiona un objeto, en nuestro caso una pequeña pelotita, y la mueve de un lado a otro, tal cual se tratase de una cadena de producción, donde

objetos van siendo modificados y trasladados a las siguientes etapas. El despliegue del prototipo se puede observar en la Fig. 21, con la HMI de la simulación sincronizada con la realidad y utilizada para SCADA de la planta.

El *software* utilizado como SCADA y HMI ha sido programado en Netlogo, donde se utiliza una librería *ad-hoc* programada en Java para comunicarse con el agente proxy, el cual se comporta como una interfaz con los sensores y actuadores de la plataforma neumática.

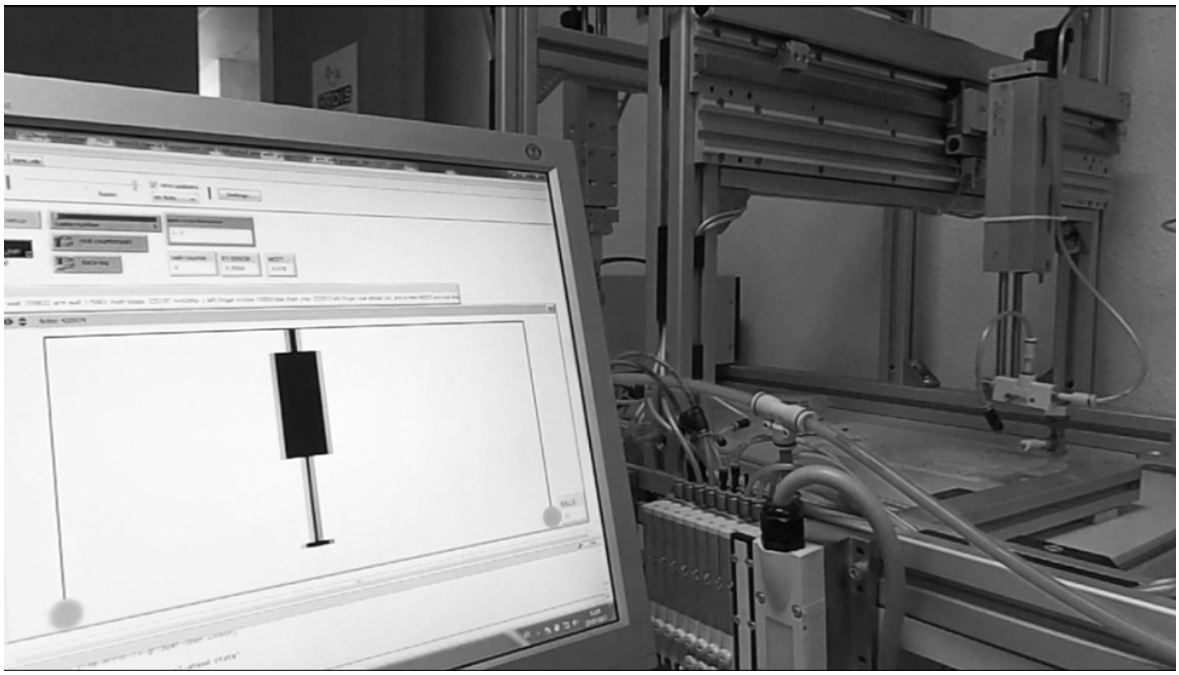


Fig. 21 Despliegue del prototipo de SCADA con HMI hacia la izquierda

El desarrollo del prototipo final de la Fig. 21 llevó 500 horas de trabajo de una persona, donde 200 horas estuvieron asignadas a la implementación del ABM en el simulador y el resto para el despliegue del SCADA sincronizado con la planta. El prototipo ha funcionado bien y puede ser visto en el video[CR].

[CR] https://www.youtube.com/watch?v=c2fC9IFZ8LU

5.2.4 CARACTERIZACIÓN DE LA PLANTA

La caracterización de la planta hace referencia a la obtención de parámetros para el modelo en el simulador que permitan reflejar la realidad con un cierto grado de precisión. En este trabajo, la única característica que ha sido tenida en cuenta es el tiempo (o velocidad): cada modelo tiene un tiempo asociado a cualquier tarea del agente.

Dado que la sincronización es realizada automáticamente mientras la planta real se encuentra en funcionamiento, esto lleva a que la primera ejecución de los pasos de la simulación los eventos simulados vaya por delante de los de la realidad.

Como el tiempo caracterizado es usado por el ABM y por el sincronizador durante la ejecución, cuanto más cercano es éste, al tiempo real medido (RT), se supone que la sincronización será mejor.

La hipótesis aquí es que la calidad de la caracterización depende de la estrategia utilizada para caracterizar, lo cual se intentará validar a continuación.

Para la caracterización de la planta existen diferentes estrategias, en este trabajo se plantearán dos muy simples, las cuales se denominan Cx1 y Cx10, las cuales son comparadas con resultados de RT.

Cx1 es el caso cuando el tiempo de caracterización es tomado únicamente la primera vez que la planta es ejecutada.

Cx10 es similar a Cx1 pero la caracterización representa el promedio de las 10 primeras ejecuciones de la planta.

Los datos caracterizados tanto para Cx1 y Cx10 son comparados con 45 ejecuciones en tiempo real (RT), que son realizadas y medidas luego de las caracterizaciones.

La Ecuación 1 Calidad de caracterización hacia la izquierda para Cx1 y hacia la derecha para Cx10 muestra cómo la calidad de la caracterización representada en porcentaje es calculada para ambos casos (Cx1 y Cx10). Q_i representa la calidad de la caracterización en promedio para cada acción/secuencia (i) realizada por el brazo robot y denotadas como *sucker on (SO), up-arm(UA), right-arm (RA), down-arm (DA), sucker off (SF), left-arm (LA)* durante $n = 45$

tareas, donde *k* denotando que la caracterización del tiempo *t* ha sido realizada antes que las medidas RT.

$$Q_i = \frac{t_{0_i}}{\frac{\sum_{k=1}^{n} RT_{k_i}}{n}} \times 100 \qquad Q_i = \frac{\frac{\sum_{k=0}^{9} t_{k_i}}{10}}{\frac{\sum_{k=10}^{n+10} RT_{k_i}}{n}} \times 100$$

Ecuación 1 Calidad de caracterización hacia la izquierda para Cx1 y hacia la derecha para Cx10

La Fig. 22 muestra la calidad de la caracterización resultante en porcentaje y en promedio para ambos casos y para cada secuencia calculada mediante la Ecuación 1 Calidad de caracterización hacia la izquierda para Cx1 y hacia la derecha para Cx10.

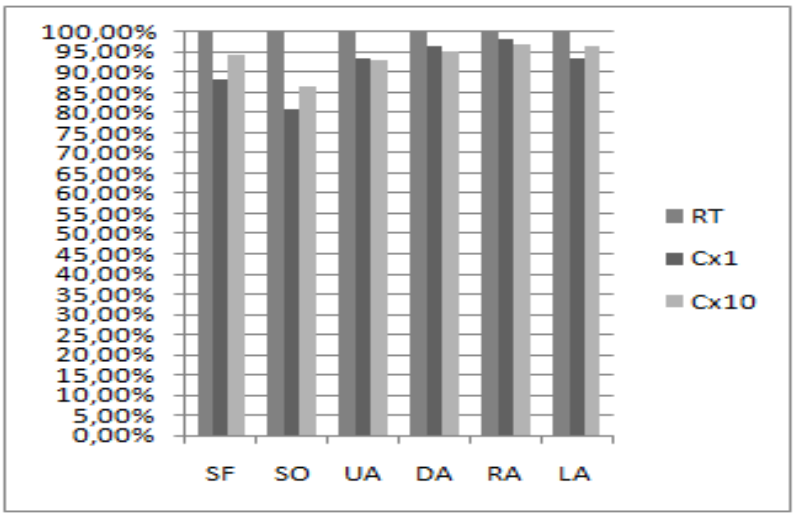


Fig. 22 Calidad de caracterización para dos estrategias

El objetivo de la Fig. 23 es dar una idea acerca del tiempo asociado a lo largo de una tarea realizada por el brazo robot. Estos valores de tiempo muestran el promedio de *real-time/on-line* medidos en segundos (fueron realizadas mientras la planta estaba en ejecución en tiempo real), para cada secuencia de las medidas de RT presentadas anteriormente. La duración de

cada tarea o ejecución es la suma del tiempo de cada secuencia asociada a la tarea, siendo de 11,743 segundos, con una resolución del *timer* de 1ms.

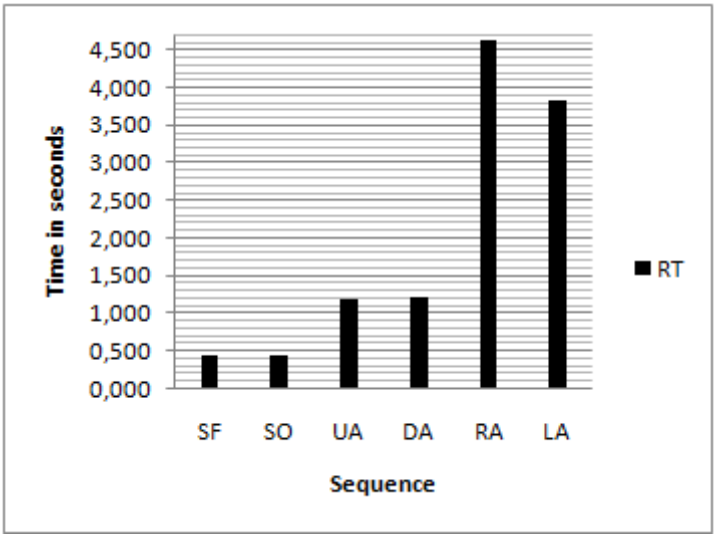


Fig. 23 Tiempo promedio por secuencia para 45 tareas RT

Como respuesta a la hipótesis planteada en esta sección, se analizan los resultados de la Fig. 22. En este trabajo, la calidad de la caracterización es mayor cuánto más cerca se encuentra del 100%, donde este valor corresponde a RT que es el tomado como referencia. Analizando visual y numéricamente la Fig. 22, se puede deducir que la estrategia Cx1 es mejor que la Cx10 para las secuencias DA y RA, mientras que Cx10 es mejor que Cx1 en las secuencias SF, FO y LA, para la secuencia UA, ambas estrategias resultan similares. Según lo analizado, no es posible usar una única estrategia para todas las secuencias y lo más lógico es que cada una sea tratada de manera particular.

En el caso de que una única estrategia deba ser utilizada para todas las secuencias, el error reportado para Cx1 es 8,23%, mientras que para Cx10, es de 6,28%, con ello esta última sería la mejor opción. En este caso, el error debe ser interpretado en función de cuán lejos está cada estrategia de RT que es la referencia.

La variabilidad a lo largo de la caracterización de cada secuencia podría ser una función de diferentes fuentes tales como la mecánica del sistema, cambios en la presión y también

asociadas a que cada secuencia recorre diferentes distancias, etc., pero este estudio no es clave en este trabajo.

En cuanto a la calidad de la caracterización, se podría decir que depende por lo menos de la resolución del *timer* utilizado, la estrategia de caracterización implementada en el lazo de control, parámetros que tendrían que ser elegidos o bien configurados dependiendo dependiendo la aplicación y, los cuales son estudiados a lo largo de este trabajo.

El siguiente apartado se enfoca en el análisis de uno de estos parámetros, el lazo de control.

5.2.5 ESTUDIO DEL LAZO DE CONTROL

En sistemas SCADA, el lazo de control determina cuán rápido el controlador puede supervisar y controlar la planta real y, para no perder información, este tiene que ser por lo menos dos veces más rápido que la velocidad a la que necesita ser supervisado y controlado el elemento más rápido de la planta.

El tiempo de cómputo del controlador, especialmente el bucle de ejecución más grande o principal está asociado con restricciones de tiempo real, las cuales dependen de los requisitos de la aplicación y pueden ser del tipo *soft, hard or critical real-time* [134]–[136]. La pregunta a contestar en este estudio es si el sistema desplegado en este trabajo podría ejecutarse en tiempo real y qué clase de restricciones de tiempo real serían las que podría alcanzar. Para ello, se realizaron una serie de ensayos y mediciones. Los ensayos consistieron en 45 tareas realizadas por el brazo robot mientras el lazo de control principal era medido. La medida relacionada con la tarea es el tiempo que le lleva cubrir cada tarea en la distancia total a recorrer. Las medidas del lazo de control consisten en contar el número de veces que se ejecuta el lazo de control principal a lo largo de las 45 tareas y dividiéndolo en diferentes ventanas de tiempo. Estas ventanas temporales representan dos límites principales, uno en 10ms. y el otro en 50ms. Los 10ms. son el valor más pequeño para considerar la medida lo suficientemente precisa, debido a que la resolución del *timer* es de 1ms. Los 50ms. representan un valor experimental que permite permite que alrededor del 1% de los casos no sean cubiertos para la distancia más pequeña relacionada con la ventosa tanto en *on* como en *off* (2mm.), cuyo valor es aceptable para esta aplicación.

La Fig. 24 muestra los resultados para los ensayos detallados anteriormente, donde el total de la distancia cubierta por una tarea es de 1426mm. y el tiempo que le lleva recorrerla es de 11,743 segundos. Por lo tanto, el recorrido de 1mm. consume 8,23ms., recordando que la menor distancia en la ruta es de 2mm. (ventosa *on/off*), entonces con estos valores este caso pésimo es cubierto por el controlador alrededor de 98,7% de las veces. El número total de lazos principales de control a lo largo de las 45 tareas ejecutadas asciende a 1683811 y únicamente 89 lazos de control principal es igual o mayor a 50ms., este tiempo representa que para unos pocos casos el error es igual o mayor a 0,5% (6mm.) para la distancia total recorrida en cualquiera de las 45 ejecuciones.

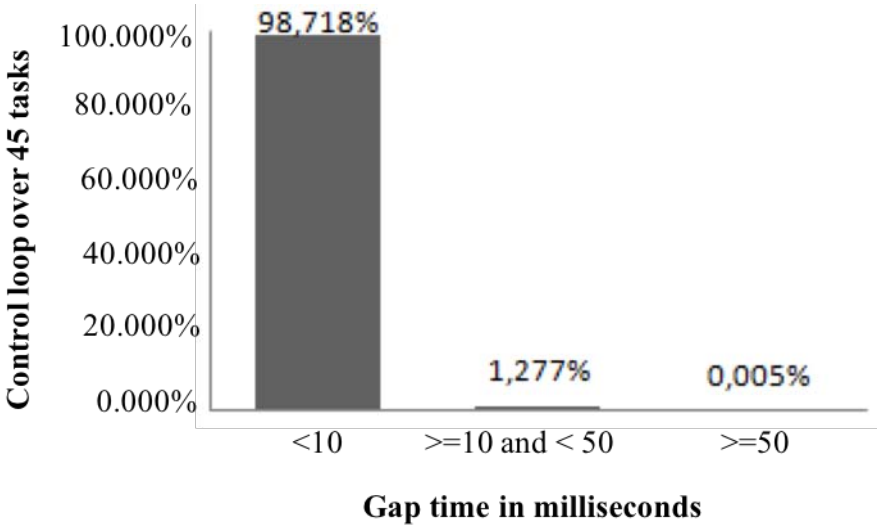


Fig. 24 Distribución del tiempo del lazo de control

Hasta aquí, los experimentos muestran que la velocidad de cómputo del controlador es suficientemente rápida para cubrir casos pésimos en cuanto a distancia (2mm.) alrededor del 98,7% de las veces, lo cual es suficientemente bueno para esta aplicación (celda de producción robotizada) y que se puede recuperar de un error sin crear un fallo total en el sistema, lo que significa que puede ser gestionada bajo restricciones de tiempo real *soft*. Estos casos de que suceden alrededor del 98,7% de las veces en este estudio son llamados casos típicos de tiempo de ejecución (TCET), cuyos valores medidos de tiempo de ejecución son capaces de cubrir el peor caso en cuanto a distancia (2mm.).

Cuando se analiza el lazo de control principal, existe un caso especial que tiene que ser tenido en cuenta, este es conocido como el peor caso en cuanto a la ejecución del tiempo del lazo principal o en inglés *Worst Case Execution Time (WCET)* [137]. El WCET es el tiempo más largo del bucle de control principal del sistema que puede suceder mientras este se encuentra en ejecución. Para que su valor se aceptable, puede estar cercano al valor temporal más grande que es aceptado por el sistema de control aunque es mejor, si se encuentra varias veces más alejado. El WCET puede ser medido de manera estática (analizando el tiempo que le lleva ser ejecutada a cada instrucción asociada al bucle de control principal), lo cual es no viable en el caso de sistemas complejos donde, las mediciones dinámicas son más común. La desventaja asociada a la medición dinámica del WCET es que durante la ejecución el valor encontrado puede corresponder al WCET que puede tener el sistema a lo largo de toda su vida y bajo las diferentes situaciones o bien puede emerger algún otro que no haya acontecido durante su estudio. Cuanto más ensayos y escenarios sean probados, la probabilidad de encontrar un nuevo WCET se incrementa, donde el número de los mismos tendría que ser determinado para cada aplicación en relación a su criticidad, sus especificaciones y sus restricciones.

En este trabajo, a lo largo de la ejecución de las 45 tareas se ha obtenido un WCET asociado a cada una de ellas, el peor de los casos de estos WCET se toma como el WCET representativo de todo el sistema.

La Fig. 25 muestra el WCET para cada una de las 45 tareas ejecutadas. Como se apuntó anteriormente, si por ejemplo se hubieran ejecutado únicamente 20 tareas, el WCET encontrado para todo el sistema sería cercano a 95ms., en la misma línea, al ejecutar 45 tareas, el WCET encontrado ha sido de alrededor de 140ms. el cual sucede durante la ejecución de la tarea 31, por lo tanto, es posible que si el número de ejecuciones de tareas se amplía, quizás se encuentre algún WCET de mayor valor al presentado en este estudio (140ms.).

Durante los 45 ensayos, la frecuencia de muestreo promedio ha sido alrededor de 3,2Khz., desafortunadamente, esto no es suficientemente rápido y el WCET encontrado es alrededor de 8,5 veces más lento que el necesitado para el peor caso a medir (2mm. ventosa *on/off*). Sin embargo, no es tan crítico como suena, dado que esta aplicación no es de tipo *hard/critical real-time* donde este tipo de situaciones que se encuentran fuera del rango de tiempo

establecido pueden generar un fallo total del sistema o algún desastre, a pesar que los 140ms. de WCET son 8,5 veces más grande, es posible la sincronización aún cuando algún bucle de control de simulación sea más largo que el requerido a nivel de campo (*field-level/shop-floor*), dado que siendo este el caso, desde el punto de vista de visualización, puede haber una sobrecompensación a nivel de simulación. Este hecho puede ser percibido en la HMI como pequeños saltos, sin embargo, estos casos en nuestro sistema tienen una probabilidad de ocurrencia baja siendo menor al 1,28% (ver Fig. 24 barras más hacia la derecha).

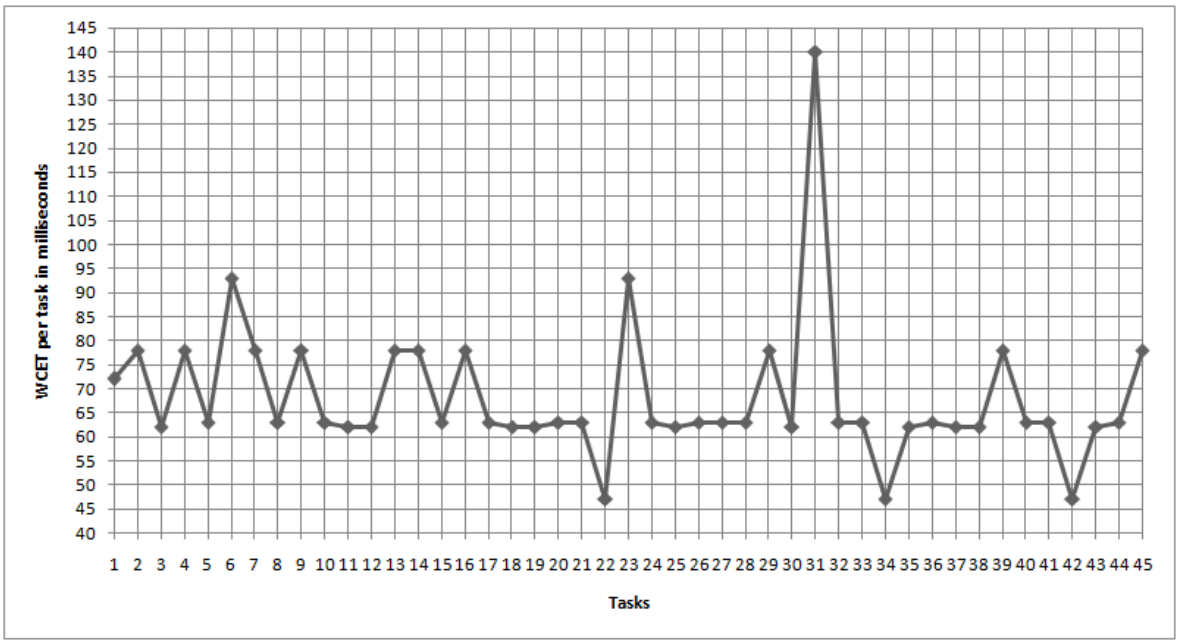


Fig. 25 WCET durante las 45 tareas

El autor de la tesis piensa que existen un gran número de aplicaciones soft real-time que se pueden beneficiar de esta experiencia/solución. En este sentido, el autor es optimista en cuanto a que es posible extender el enfoque ALOHA a aplicaciones tipo *hard real-time* pero, el marco de trabajo debería ser ligeramente modificado tanto en hardware como en software.

5.2.6 SINCRONIZANDO LA SIMULACIÓN ABM CON LA REALIDAD

El middleware de sincronización (SYM) provee una representación de la realidad en tiempo real del ABM y actúa también como mecanismo de control, en el cual los eventos de la realidad

son comparados con los esperados en la simulación ABM. Más aún, si existen considerables diferencias entre el ABM y la planta real, hay mecanismos de corrección que pueden ser aplicados, dado que estas diferencias afectarán significativamente la representación de la planta real en la HMI.

El objetivo de esta sección es la de estudiar la calidad de la sincronización del enfoque ALOHA para cada tarea realizada por el brazo robot, para cada secuencia que integra cada tarea de una manera más detallada el estado que más interesa, in-time. Para ello, se analiza la calidad con la que el módulo SYM del enfoque ALOHA es capaz de sincronizar.

Los datos a analizar provienen de medidas mientras el sistema se encuentra bajo ejecución (simulación concurrente con la realidad). La calidad de la sincronización se determina mediante el número de *hard real-time synchronization events* (HSE) que son contabilizados en cada estado (*in-time*, *real-ahead*, *real-behind*). Este número para representar la calidad de sincronización es representado en porcentaje de cada estado sobre el total de HSE.

Mientras una tarea es ejecutada, hay diferentes secuencias que están asociadas a ella, es por ello que el primer estudio está enfocado en determinar la calidad de sincronización en cada secuencia.

Como resultado, la Fig. 26 muestra el grado de sincronización en promedio para cada secuencia a lo largo de 45 tareas ejecutadas. Los estados del sincronizador son cuatro, *in-time*, *real-ahead*, *real-behind* and *wait*, pero el último no es mostrado dado que puede ser deducido, ya que la suma de los cuatro estados debe ser 100%.

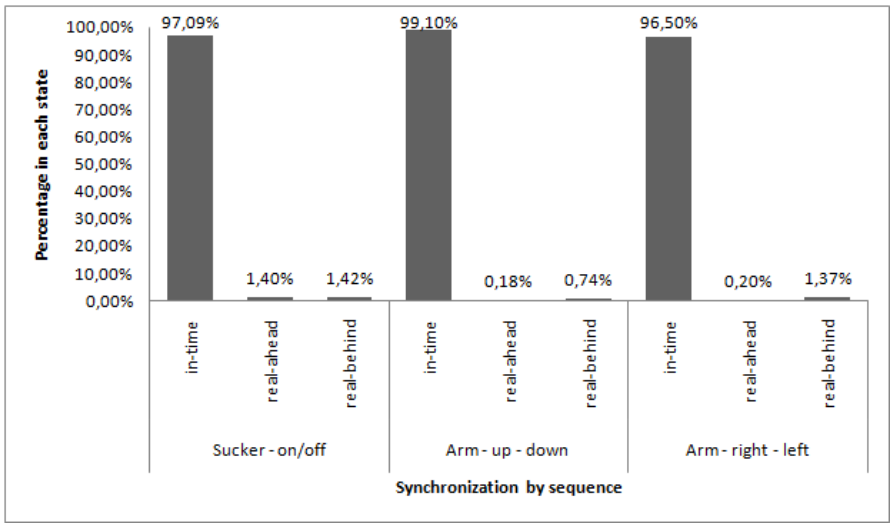


Fig. 26 Sincronización por secuencia de los diferentes estados

El mejor estado del sistema o el estado buscado es el in-time, donde la simulación se encuentra sincronizada con la realidad y cuyo valor debería ser lo más cercano posible al 100%, donde en ese caso significaría que el simulador ABM ha estado sincronizado todo el tiempo con la planta real. El segundo mejor estado es el *real-behind*, dado que en este estado la realidad va más lenta que la simulación y, en este caso, el ABM tiene que esperar a la realidad. El peor caso es en el estado *real-ahead*, debido a que la simulación ha ido más lenta que la realidad y entonces la realidad debería esperar al modelo (esto de ser posible no debería ocurrir y si así fuere se puede configurar y dicha espera tendría que ser casi imperceptible), el problema de este caso es que introduce un retardo innecesario en el comportamiento de la planta real. Estos dos últimos estados (*real-ahead* y *real-behind*), debería ser lo más cercanos posible a 0%, lo cual significaría que la mayor parte del tiempo el sistema ha estado sincronizado (*in-time*). Para resumir, los problemas de sincronización son más perceptibles visualmente a medida que el estado *in-time* se aleja del 100%.

Las consecuencias de no tener el estado in-time cercano a 100% se percibe como pequeños retardos o pequeños saltos en la HMI (simulación), o en la realidad viendo al componente bajo sincronización esperando mientras la simulación intenta llegar al mismo punto en el entorno simulado.

Como el interés es el de tener al sistema sincronizado, representado por el estado in-time, es interesante saber que para este estado en cada secuencia cuales son los máximos y mínimos valores así también como el promedio.

Como resultado, estos valores son mostrados en la Fig. 27 para cada secuencia de un promedio de las 45 tareas. Se puede apreciar que el valor más bajo en cuanto a la calidad de la sincronización en el estado in-time sucede para la secuencia asociada con la ventosa siendo alrededor del 84%.

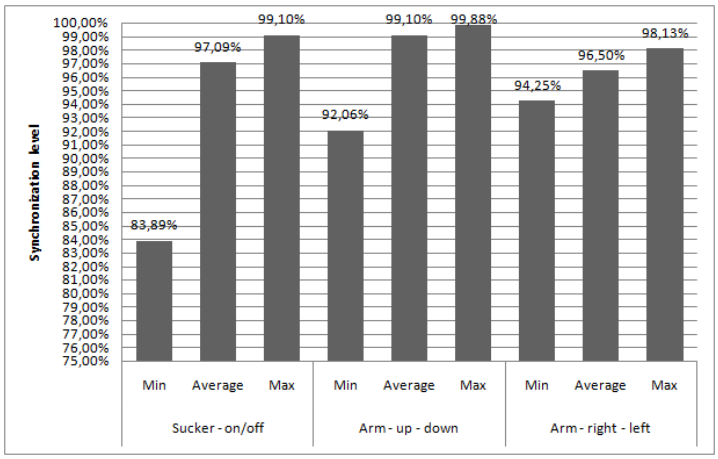


Fig. 27 Mín., máx. y promedio del estado in-time

Mientras que en la Fig. 27 se muestra información para cada secuencia, la Fig. 28 explica la sincronización a nivel global para las 45 ejecuciones, como resultado, se muestran los valores que representan calidad de sincronización a nivel global para el promedio de las 45 ejecuciones en cada uno de los estado y en especial en el estado de interés (in-time) se pueden apreciar sus valores máximos y mínimos.

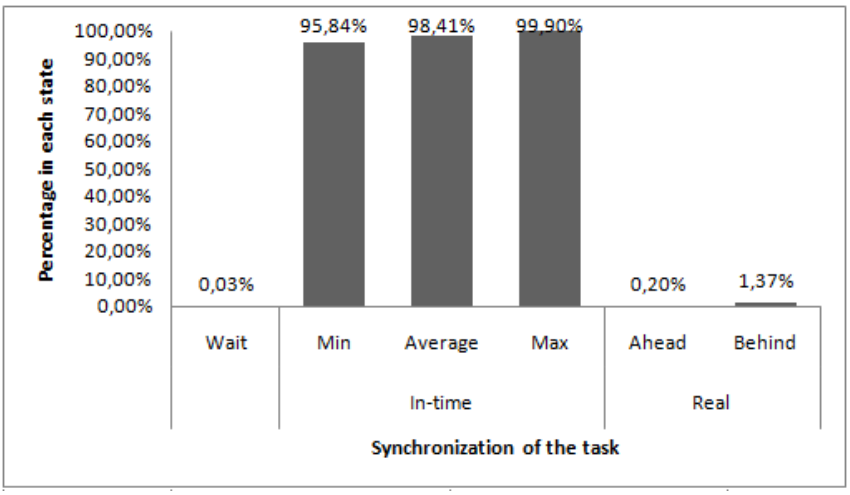


Fig. 28 Sincronización durante 45 tareas

Comparando la Fig. 17 y la Fig. 28, es posible observar que la sincronización sobre una tarea es mejor que la contribución individual de cada secuencia, este hecho se corresponde con la sensación de sincronización al ver la representación de la planta en la HMI.

Como resultado global de este estudio de sincronización, se puede decir que ha funcionado suficientemente bien tomando como base tanto los valores cuantitativos mostrados en esta sección así como cualitativamente a través de la sensación percibida por el usuario al observar la HMI.

5.2.7 CONCLUSIONES

Durante esta sección se ha mostrado la factibilidad de utilizar un simulador ABM como SCADA en el área industrial, para ello se utilizó una plataforma robotizada neumática que representa una celda de producción. Se ha analizado mediante simulación que es factible supervisar y controlar unos 20 brazos robots de estas características con la tecnología utilizada en esta tesis. Se ha medido y analizado la caracterización para dos estrategias diferentes, el lazo de control y la calidad de la sincronización tanto de manera global como por cada secuencia.

La implementación software del prototipo se realizó utilizando el enfoque ALOHA y el despliegue hacia la plataforma siguiendo la metodología/diagrama de flujo propuesto para este enfoque.

Con todo, se espera en el futuro poder probar el *framework* de esta tesis con otras aplicaciones y extenderlo a soluciones tipo *hard real-time*.

En el siguiente capítulo se presentan las conclusiones de esta tesis y sugerencias sobre trabajos futuros.

6 Conclusiones y trabajo futuro

6.1 Objetivo

Esta tesis aborda la temática de cómo se pueden utilizar simuladores ABM durante el proceso que va desde el diseño hasta el despliegue en entornos industriales, dejando el mismo para SCADA en tiempo real de la solución final. El objetivo de este enfoque es el de estudiar si es posible reducir costes no recurrentes de ingeniería y agilizar el citado proceso. Para poder probar que esto es factible se desarrolló un *framework* y se implementaron dos prototipos que representan aplicaciones de transporte de objetos.

La pregunta de investigación conduce a lo largo de esta tesis a analizar qué métodos y herramientas deberían desarrollarse para proveer de un marco de trabajo genérico, factible de implementar, adaptable fácilmente a diferentes aplicaciones y que provea agilidad en el despliegue de soluciones en el área industrial y, al mismo tiempo, cumpla con los requerimientos actuales y tendencias futuras de este ámbito.

Para mostrar la validez y utilidad de la pregunta de investigación se destacaron los desafíos actuales y las tendencias a futuro que enfrentan y enfrentarán las empresas del sector industrial. Como punto importante de coincidencia se presenta la necesidad de ser cada vez más competitivas en un entorno que requiere una rápida adaptación y agilidad al momento de implementar nuevas aplicaciones, lo cual puede llegar a ser posible a través de nuevos enfoques/herramientas y la aplicación de paradigmas con mayor poder de abstracción que ayuden a hacer frente a estos problemas.

La revisión histórica presentada de los programas de simulación y los diferentes paradigmas nombrados muestran que en el área industrial, en el primer caso, su uso está mayormente limitado a la etapa de diseño, siendo dejados de lado en la etapa de despliegue (*shop-floor*), en el segundo caso se observa la resistencia en este entorno al uso de paradigmas tales como el de agentes y holones, escepticismo sustentado en la baja confianza y en la escasa capacitación en estos paradigmas al mismo tiempo que la necesidad de un mayor número de casos de éxito con estas tecnologías. En esta tesis en este sentido se muestra que es factible acercar el mundo

de la simulación hacia el *shop-floor* contribuyendo a incrementar los casos de éxito en el uso de estas tecnologías y paradigmas a nivel industrial, a la vez que permite agilizar el despliegue de sistemas industriales.

6.2 Trabajo realizado

La pregunta de investigación ha sido respondida proponiendo y desarrollando una serie de herramientas y aplicándolas a dos prototipos. Entre las herramientas se puede destacar el enfoque/arquitectura ALOHA que permite unir la etapa de diseño y la de despliegue, convertir componentes del *shop-floor* en agentes y sincronizar el mundo simulado con el real. ALOHA aparte de ser un acrónimo, está inspirado en que los actos comunicacionales entre agentes 'educados' deberían iniciarse y terminar mediante formas simples de saludos, tal cual lo hacen normalmente las personas.

Llevar a cabo lo planteado en esta tesis requiere de una amplia gama de conceptos y tecnologías base, las cuales han sido analizadas y detalladas para permitir al lector una mejor y más rápida comprensión de todo lo que hay como base en cuanto a las herramientas y los prototipos de este trabajo.

La validez de las herramientas desarrolladas en el marco de esta tesis han sido probadas desde dos ángulos diferentes, el primero que muestra que es factible implementar la solución propuesta y el segundo verifica si el sistema es adecuado o sirve al propósito para el que fue desarrollado. La factibilidad se demuestra a través del despliegue de los dos prototipos y si el sistema es adecuado o sirve para su propósito mediante el análisis de diferentes parámetros tales como calidad de sincronización, restricciones de tiempo real y el funcionamiento del sistema bajo diferentes escenarios.

Desde este punto en adelante se presentan diferentes resultados obtenidos en esta tesis y que se resumen a modo de conclusiones como:

- Contribuciones científicas y técnicas
- Prototipos
- Publicaciones

6.2.1 CONTRIBUCIONES CIENTÍFICAS Y TECNOLÓGICAS

Las contribuciones se presentan en dos grupos bien diferenciados, el primero compuesto por aquellas que incluyen conceptos considerados más originales y el segundo que incluye las contribuciones que a pesar de ser relevantes están más relacionadas con aspectos tecnológicos.

Las contribuciones de carácter conceptual son las que en esta tesis se denominan como las herramientas que integran el marco de trabajo o *framework* y se dividen en cuatro grupos que han sido explicados en detalle en el capítulo 4 y se presentan a continuación como parte de los resultados.

El conjunto integrado por **el modelo ABM y la arquitectura de control** distribuida sirve para representar globalmente y de manera genérica sistemas de la realidad que se desean simular y al mismo tiempo controlar. En el primer caso los agentes del sistema se dividen en dos tipos, aquellos específicos a la aplicación y los llamados ALOHA que confieren características de agente a componentes de la realidad. En el segundo caso el control se organiza de manera distribuida tanto para los agentes únicamente simulados como para aquellos que por su naturaleza híbrida (los agentes ALOHA) poseen un componente de la realidad y su contraparte simulada.

El enfoque ALOHA permite conferir las características de los agentes a componentes de la realidad, organizando sus elementos a través de una arquitectura de tres niveles que ayuda con la comunicación interna y externa a los agentes tipo ALOHA, al mismo tiempo, la capa intermedia (módulo de sincronización) es el responsable de mantener sincronizado el modelo en simulación con su contraparte real.

El módulo de sincronización es necesario para crear la capa intermedia entre el nivel *H* y el *L* en los agentes tipo ALOHA y permite la comunicación entre ambos. Es el responsable de la sincronización de la simulación con la realidad tanto para los denominados eventos HSE y SSE.

Con todo, **la metodología ALOHA** integra lo anterior para establecer los procedimientos para la implementación de sistemas basados en agentes de tipo ALOHA y que utilizan simuladores ABM para SCADA de la realidad, desde la etapa de diseño hasta el despliegue de los mismos.

Por otra parte, las contribuciones tecnológicas tienen importancia desde el punto de su aplicación a nivel industrial, constituyendo una fuente de viabilidad en este sentido y se dividen en dos tipos.

Aplicación de la tecnología de agentes al entorno industrial. Si bien existen en el estado del arte ejemplos de aplicación de agentes en el entorno industrial ya sean de simulaciones puras o entornos industriales controlados por *frameworks* basados en agentes, estos casos de éxito no son suficientes para ganar la confianza del ámbito industrial y en consecuencia que se utilice este paradigma de manera más masiva. Es por ello que los dos casos de aplicación de esta tesis y las herramientas aquí desarrolladas e implementadas contribuyen a aumentar los casos de éxito y a ampliar la manera en la que la tecnología de agentes se utiliza a nivel industrial.

Acortar las distancias entre el diseño y el despliegue de sistemas industriales. Existen diversos motivos por los cuales los programas de simulación han sido utilizados mayormente para la etapa de diseño verificación y validación y luego al ser desplegados un proceso de re-ingeniería es llevado a cabo para terminar supervisando y/o controlando el sistema desplegado por tecnologías diferentes a la de los simuladores. En algunos casos, por ejemplo, en sistemas de tiempo real críticos, al momento de ser escrita esta tesis, esto parece inevitable. También, parece ser, que existe un rango de aplicaciones, como por ejemplo las de *soft real-time* y quizás algunas de *hard real-time* en la que la brecha entre diseño y despliegue puede ser acortada, manteniendo la simulación durante todo el proceso, 'finalizando' este con la supervisión y/o control sobre la aplicación final que es ejecutada desde el simulador. En este sentido, tecnológicamente se ha mostrado que es posible, validando esta afirmación por medio de dos prototipos a nivel industrial del tipo *soft real-time*.

6.2.2 PROTOTIPOS

Los siguientes prototipos han sido desarrollados e implementados a lo largo de esta tesis:

Sistema de transporte de muestras con AGVs. Ha sido el primer validador utilizado en esta tesis para probar que es posible utilizar simuladores ABM como SCADA de entornos industriales. Ha servido para desarrollar y probar las herramientas que componen el *framework*

de esta tesis, en especial, en este caso se valida la sincronización del tipo HSE y SSE. Ha permitido demostrar que la simulación se puede ejecutar en tiempo real y de manera concurrente con la realidad, permitiendo también integrar entornos de realidad mixta.

Celda de producción con brazo robot neumático. Ha servido como segundo demostrador de que es factible utilizar la simulación ABM como SCADA de la realidad en aplicaciones industriales. Se ha utilizado para implementar la solución utilizando la metodología propuesta y se ha implementado la sincronización del tipo HSE, con un algoritmo mejorado. También se ha usado para medir el tiempo invertido desde el diseño hasta el despliegue de la solución completa, permitiendo validar el *framework* como una solución que conduce a agilizar el despliegue de sistemas industriales.

6.2.3 PUBLICACIONES

Esta tesis ha dado como resultado nueve trabajos, de los cuales siete están publicados y dos pendientes de aceptación y publicación; de los cuales seis pertenecen a congresos tanto nacionales como internacionales y tres a revistas científicas, los cuales se citan brevemente a continuación:

En [106] se analizan diferentes herramientas de software que permiten plantear un entorno adecuado para sistemas internos de transporte que utilizan mini AGVs, se implementa una primera versión simulada con los datos planteados. En [138] se propone cómo se puede transformar un laboratorio de análisis clínicos convencional centralizado en un sistema multiagente, de esta manera distribuir el control y hacer las instalaciones más adaptables a análisis de muestras con prioridad y otros eventos. En [123] se demuestra que se pueden utilizar AGV en vez de cintas transportadoras para el traslado de muestras, obteniendo un rendimiento equivalente en cuanto a cantidad de muestras procesadas, pero con la ventaja de que con los AGV se confiere mayor adaptabilidad, flexibilidad y robustez. En [122] se presenta el modelo basado en agentes que utiliza AGVs para el transporte interno en laboratorios automatizados. En [139] se muestra la arquitectura del controlador basada en sistemas multi-agentes para el transporte de objetos mediante AGVs. En [140] se presentan las herramientas que constituyen el entorno de trabajo como una solución genérica, que se puede llevar a cabo con cualquier simulador ABM. En [141] se presenta un trabajo donde se muestra cómo se

pueden desplegar prototipos a escala, de sistemas de transporte que utilizan robots. En [142] se muestra cómo se pueden ejecutar simulaciones ABM sincronizadas con la realidad, para el control de sistemas de transporte. Finalmente, en [143] se presenta una arquitectura de control basada en simuladores de agentes para un rápido desarrollo de sistemas multi robots.

6.3 Conclusiones

El entorno industrial necesita por un lado de tecnologías (paradigmas, herramientas *software*, *hardware*, etc.), así como de 'nuevas' formas de integración, todo ello enfocado a poder hacer frente a los desafíos actuales y futuros de esta área, siendo algunos de ellos la flexibilidad, la robustez, la evolutividad, la adaptabilidad, la eficiencia, la incorporación del I2oT (de *Industrial Internet of Things*), las VF (de *Virtual Factories*), las SF (de *Smart Factories*), etc. y, por el otro lado, de demostradores que permitan poner a prueba la dupla tecnologías y desafíos, en contextos similares al industrial, con el fin de que éste pueda ganar confianza en el uso de 'nuevas' tecnologías (y sus diferentes maneras de integrarlas) como posibles fuentes de soluciones. A continuación, y a modo de conclusiones de este capítulo, se citan en tres categorías diferentes de los resultados del trabajo realizado, los cuáles intentan contribuir en algunas de las direcciones ya mencionadas, siendo:

1) creación de un marco de trabajo integrado por modelos, arquitecturas y una metodología, que representan una solución genérica, de cómo llevar a cabo supervisión control y adquisición de datos desde simuladores ABMs.

2) se ha implementado tanto en *software* como en *hardware* dos prototipos que han permitido verificar y validar la hipótesis de esta tesis así como las herramientas desarrolladas y citadas en el capítulo 4.

3) las publicaciones en congresos tanto nacionales [106] como internacionales [122], [123], [138], [139], [143], que han permitido exponer los resultados obtenidos y, en consecuencia, recibir el *feedback* de los revisores, con el objetivo de para mejorar la calidad del trabajo y, adicionalmente, una vez aceptadas, han constituido un elemento de validación por parte de la comunidad científica. Dentro de las publicaciones en revistas científicas con factor de impacto, a fecha de escritura de esta tesis se posee una aceptada [140] y dos [141] y [142] pendientes de aceptación.

A continuación se presentan algunas sugerencias que podrían resultar interesantes para futuras investigaciones.

6.4 Sugerencias para futuras investigaciones

Dado que todo trabajo de tesis no puede ser considerado como completamente finalizado, el mismo durante el proceso ha servido como semillero para nuevas ideas, que podrían ser interesantes para futuras investigaciones/tesis. Es por ello que a continuación se citan en los siguientes párrafos los aspectos más importantes que podrían ser investigados:

1. **Balance entre supervisión y sincronización.** Cuando se representa el modelo de una planta mediante una simulación sincronizada con la realidad, el usuario observa la ejecución de la simulación y no datos directos de supervisión. Realizar la representación de esta manera parece conducir a la descongestión del canal de comunicaciones de datos de supervisión, lo cual podría ser muy útil en sistemas donde se paga por el uso del canal de comunicaciones. Un correcto balance entre supervisión y sincronización permitiría reducir costes sin interferir perceptiblemente en lo que ve el usuario.
2. **Uso de la sincronización como herramienta de diagnóstico.** Se postula que al intentar llevar la simulación el mismo ritmo de la realidad, el registro y análisis del desfasaje de la simulación con la realidad podría hacer presumir, fallos o degradación de componentes del sistema y en consecuencia se podría actuar correctivamente o preventivamente.
3. **Mejorar/incrementar los métodos de sincronización.** En esta tesis, se han implementado dos métodos (EDM e ISM) que actúan ante 'errores' o desfasajes en la sincronización. Sería interesante mejorar los mismos y/o proveer al sincronizador de características adicionales que permitan aplicar diferentes estrategias que se adapten a los requerimientos de concurrencia entre simulación y realidad de diferentes clases de aplicaciones.
4. **Analizar la extensibilidad de esta tesis a aplicaciones *hard real-time*.** En esta tesis se ha validado la utilización de simulación ABM para SCADA de la realidad en aplicaciones industriales *soft real-time*. Sería interesante estudiar la aplicabilidad de este trabajo en aplicaciones *hard real-time.* Para ello, es muy probable las tecnologías tanto

software como *hardware* deban ser adaptadas a las restricciones de tiempo real de las aplicaciones de este tipo.

5. **Crear una librería con agentes tipo ALOHA.** Tener disponible una librería con agentes tipo ALOHA es otra manera de ayudar al diseñador a implementar más rápidamente soluciones en la que se necesitan convertir componentes de la realidad en agentes. Con ello el proceso de creación con este enfoque se convierte en un proceso de adaptación/parametrización, siendo esto más rápido que implementar el agente ALOHA desde cero.
6. **Implementar módulos para generación automática de código.** Generar automáticamente el código a embeber (transformación de L_V a L_R) en los componentes de la realidad puede ayudar a agilizar aún más el proceso de despliegue, dado que contribuiría a minimizar los errores introducidos durante la programación y la necesidad de conocer el lenguaje de programación de los dispositivos embebidos.

Con todo, este trabajo se constituye como una base sólida no solo para construir SCADA de plantas que permitan un tiempo real flexible a determinado nivel sino también de otras investigaciones en el ámbito del desarrollo de SCADA de acuerdo con las tendencias actuales en el ámbito industrial, del transporte y de la gestión de infraestructuras.

7 Conclusions and future work

7.1 Objective

This thesis addresses the issue of how can be used ABM simulators during the process from design to deployment in industrial environments, leaving it for real-time SCADA of the final solution. The aim of this approach is to study whether it is possible to reduce non-recurring engineering costs and speed up that process. In order to prove that this is feasible a framework has been developed and two prototypes representing objects transportation applications have been implemented.

The research question leads along this thesis to analyze what methods and tools should be developed to provide a generic framework, feasible to implement, easily adaptable to different applications and to provide flexibility in the deployment of solutions in the industrial area and at the same time, meet current requirements and future trends in this area.

To demonstrate the validity and usefulness of the research question the current challenges and future trends facing and will face the industrial companies were highlighted. As an important point of coincidence between companies is the need to be increasingly competitive in an environment that requires rapid adaptation and flexibility when implementing new applications, which can be possible through new approaches/tools and the application of paradigms with more power of abstraction that help to face these problems.

The historical review presented about simulation programs and the different paradigms appointed show that in the industrial area, in the first case, their use is mostly limited to the design stage, being left behind in the deployment phase (shop-floor), in the second case the resistance is observed in this domain when paradigms such as agents and holons are proposed as technological solutions, skepticism supported by low confidence and poor training in these paradigms while the need for a greater number of success cases with these technologies. This thesis in than sense shows that it is feasible to bring the world of the simulations to the shop-floor helping to increase the success stories in the use of these technologies and paradigms at industrial level, while speeds up deployment of industrial systems.

7.2 Work done

The research question has been answered by proposing and developing a series of tools and applying them to two prototypes. Among the tools ALOHA approach/architecture can be highlighted that allows to join the stages of design and deployment, turning shop-floor components into agents and synchronize simulation with reality. ALOHA besides being an acronym, is inspired by the communicative acts between 'educated' agents should begin and end with simple forms of greetings, as is usually done by people.

Carry out that issues raised in this thesis requires a wide range of concepts and core technologies, which have been analyzed and detailed to enable the reader better and faster understanding of all that is as basis regarding to the tools and prototypes of this work.

The validity of the tools developed in the framework of this thesis have been tested from two different perspectives, the first one to show that it is feasible to implement the proposed solution and the second one checks whether the system is adequate or serves the purpose for which it was developed. The feasibility is demonstrated through the deployment of two prototypes and whether the system is adequate and serves its purpose by analyzing various parameters such as synchronization quality, real-time constraints and testing the system under different scenarios.

From this point onward different results obtained in this thesis are presented and summarized as conclusions such as:

- Scientific and technological contributions
- Prototypes
- Publications

7.2.1 SCIENTIFIC AND TECHNOLOGICAL CONTRIBUTIONS

Contributions are introduced in two distinct groups, the first ones comprising those including most original considered concepts and the second ones include the contributions despite being relevant are more related to technological aspects.

Conceptual contributions are called in this thesis as the tools that make up the framework and are divided into four groups which have been explained in detail in Chapter 4 and are presented below as part of concluding results.

ABM model and control architecture. The set composed by the ABM model and the distributed control architecture serves to represent globally and generically reality systems to be simulated and controlling at the same time. In the first case the system agents are divided into two types, those specific to the application and the other ones called ALOHA used for conferring agent characteristics to components of reality. In the second case control it is distributed organized both agents that are simulated only as well as for those with hybrid nature (ALOHA agents) due that they have a component of reality and their simulated counterpart.

ALOHA approach. It allows to confer agent characteristics to components of reality, organizing its entirety through a three-tier architecture that helps with internal and external communication to the ALOHA type agents, while the intermediate layer (synchronization module) it is responsible for maintaining simulation model synchronized with its real counterpart.

Synchronization module. It is the intermediate layer between the high level (*H*) and the low level (*L*) in ALOHA type agents and allows communication between them. It is responsible also for synchronizing simulation to reality for both events called hard real-time synchronization events (HSEs) and soft real-time synchronization events (SSEs).

ALOHA methodology. It suggests how should implement systems integrated by agents of ALOHA type that use ABM simulators as SCADA of reality, from the design stage to their deployment.

On the other hand, technological contributions are important from the point of application to industrial level, constituting a source of viability in this regard and are divided into two types.

Application of agent technology to the industrial environment. While there are in the state of the art examples of application of agents in industrial environments whether pure simulations or industrial environments controlled by agent-based frameworks, these success cases are not enough to earn the trust of the industrial sector and consequently use this paradigm more massively. That is why the two cases of application of this thesis and tools

developed and implemented here help to increase the success stories and expand the way in which agent technology is used in the industrial domain.

Bridge the gap between design and deployment of industrial systems. There are several reasons why simulation programs have been mostly used for design stage, verification and validation and then for deployment process re-engineering is carried out for monitoring and or controlling by using different technologies than simulators. In some cases, for example in critical real-time systems, at the time of writing this thesis, it seems inevitable. Also, it appears that there is a range of applications such as the soft real-time and perhaps some hard real-time in which the gap between design and deployment can be shortened, maintaining simulation throughout all the process, 'ending' the supervision and or control over the final application running from the simulator. In this sense, it has been shown that it is technologically possible, validating this claim by two industrial prototypes of soft real-time type.

7.2.2 PROTOTYPES

Next prototypes have been developed and implemented along this thesis:

1. **Sample transport system with AGVs.** It was the first validator used in this thesis to prove that simulators can use ABM as SCADA of industrial applications. It has served to develop and test the tools that make up the framework of this thesis, particularly, in this case synchronization of type HSE and SSE have been validated. It has demonstrated that the simulation can be run in real time and concurrently with reality while also allowing the integration of mixed reality environments.
2. **Production cell with pneumatic robot arm.** It has served as a second demonstrator that is feasible to use ABM simulations as SCADA of reality in industrial applications. It has been used to implement the solution using the methodology proposed and it has implemented synchronization of HSE type with an improved algorithm. It has also been used to measure the time spent from the design to the deployment of a complete solution, allowing validation of the framework as a solution leading to speed up deployment time of industrial systems.

7.2.3 PUBLICATIONS

This thesis has resulted in nine papers, of which seven are published and two pending of acceptance and publication; six of which belong to both national and international conferences and three to scientific journals, all of them are briefly mentioned below:

In [106] different software tools that allow pose a suitable environment for internal transport systems that use mini AGVs are analyzed and a first simulated version is implemented. In [138] it suggests how can be transformed a conventional centralized clinical analysis laboratory on a multi-agent system thus distribute control and make the facilities more adaptable to priority sample analysis and other events. In [123] it is shown that AGVs can be used instead of conveyors for transporting samples, obtaining an equivalent performance in terms of number of samples processed, but with the advantage that AGVs confer greater adaptability, flexibility and robustness. In [122] an agent-based model that uses AGVs for internal transport in automated laboratory model is introduced. In [139] it is shown the controller architecture based on multi-agents systems for transportation objects by using AGVs. In [140] are shown some tools constituting a framework which can be used as a generic solution that can be performed with any ABM simulator. In [141] a work that shows how you can deploy prototypes to scale transport systems using robots is presented. In [142] is shown how ABM simulations can be run in sync with the reality for controlling transport systems. In [143] is introduced a control architecture based on agent simulators for rapid development of multi robot systems.

7.3 Conclusions

The industrial environment needs in one hand of technologies (paradigms, software tools, hardware, etc.) as well as 'new' forms of integration of them, all focused to cope with current and future challenges in this area, which some them are flexibility, robustness, scalability, adaptability, efficiency, incorporating of I2T (Industrial Internet of Things), VF (Virtual Factories), SF (Smart Factories), etc. and, on the other hand, allowing demonstrators to test the technologies and challenges duo in similar contexts to the industrial one, so that it can gain confidence in the use of 'new' technologies (and their different ways of integrating) as possible sources of solutions. Then, by way of conclusion of this chapter, the work carried out is

mentioned through three different categories, which attempt to contribute in some of the aforementioned address, where:

1) creation of a framework composed by models, architectures and methodology representing a generic solution of how to carry out monitoring and control data acquisition from ABMs simulators.

2) has been implemented in both software and hardware two prototypes that have allowed to verify and validate the hypothesis of this thesis and the tools developed and cited in Chapter 4.

3) some works have been published in national [106] and internationals [122], [123], [138], [139], [143] congress, in order to present the results and receive feedback from reviewers, to improve the quality of work and, therefore, once accepted they have been an element of validation by the scientific community. In addition, some publications in scientific journals with impact factor have been written, one of them has been accepted [140] and two [141] and [142] are pending of acceptance.

Then, some suggestions that may be of interest for future research are introduced.

7.4 Suggestions for future works

Since the entire thesis work cannot be considered completely finished, the same during the process has served as a breeding ground for new ideas, which could be interesting for future research/thesis. That is why then are cited in the following paragraphs the most important aspects that could be investigated:

1. **Balance between supervision and synchronization**. When the model of a plant is represented through a simulation synchronized with reality, the user sees the simulation running but not a representation from direct monitoring data. Making representation in this way it seems to lead to decongest the communication channel of monitoring data, which could be very useful in systems that there is a payment for using the communication channel. A proper balance between monitoring and synchronization would significantly reduce costs without interfering significantly in the user experience.

2. **Using synchronization as a diagnostic tool**. It is postulated that when trying out simulation pace with reality, registering and analyzing of simulation mismatch with reality could boast failure or degradation of system components and therefore could act correctively or preventively.
3. **Improving/increasing synchronization methods**. In this thesis, have been implemented two methods (EDM and ISM) acting to 'errors' or lags in timing. It would be interesting to improve them and or provide the synchronizer additional features that allow different strategies to suit the requirements of concurrency between simulation and reality of different kinds of applications.
4. **Analyze the extensibility of this thesis to hard real-time applications.** This thesis has been validated using an ABM simulation for SCADA of reality in soft-real-time industrial applications. It would be interesting to study the applicability of this work to hard real-time applications. It is therefore likely both software and hardware technologies must be adapted to the real-time constraints of this types of applications.
5. **Create a library of ALOHA alike agents.** Having available a library with this type of agents is another way to help the designers to implement solutions that are needed to convert components of reality into agents faster. This creation process with this approach becomes a process of adaptation/parameterization, this being faster than creating ALOHA agent from zero.
6. **Implementation of modules for automatic code generation.** Automatically generation of code to embed (transformation of L_V into L_R) into components of reality can help further to speed up the deployment process, as it would contribute to minimize errors introduced during programming and the need to know the programming language of embedded devices.

Anexo A – Distribución del código en los casos de estudio

El objetivo de este anexo es mostrar la distribución del código para las aplicaciones utilizadas como demostradores en esta tesis. Dividiendo el código de cada caso de estudio en diferentes grupos y analizando el peso relativo que estos tienen respecto al total, tanto para la aplicación completa como para el agente ALOHA. Con ello, sería posible inferir hacia donde se deberían dirigir los esfuerzos para mejorar la solución propuesta en esta tesis a nivel global y a nivel particular hacia direcciones que permitan un despliegue más ágil, verificaciones y validaciones a nivel de simulación y a nivel de componentes del *shop-floor* más rápidas y robustas, etc.

A.1 Sistema de transporte con AGVs

La aplicación para el transporte de muestras en laboratorios de análisis clínicos presenta una distribución relativa del código respecto al total tal como se presenta en la Fig. 29. Se puede observar que el código para los agentes ALOHA representa el mayor porcentaje ya que incluye también el código embebido en el componente de la realidad. A su vez en la Fig. 30 se presenta la distribución relativa a los diferentes elementos que componen la arquitectura del agente ALOHA siendo en este caso el código en el dispositivo embebido el de mayor peso.

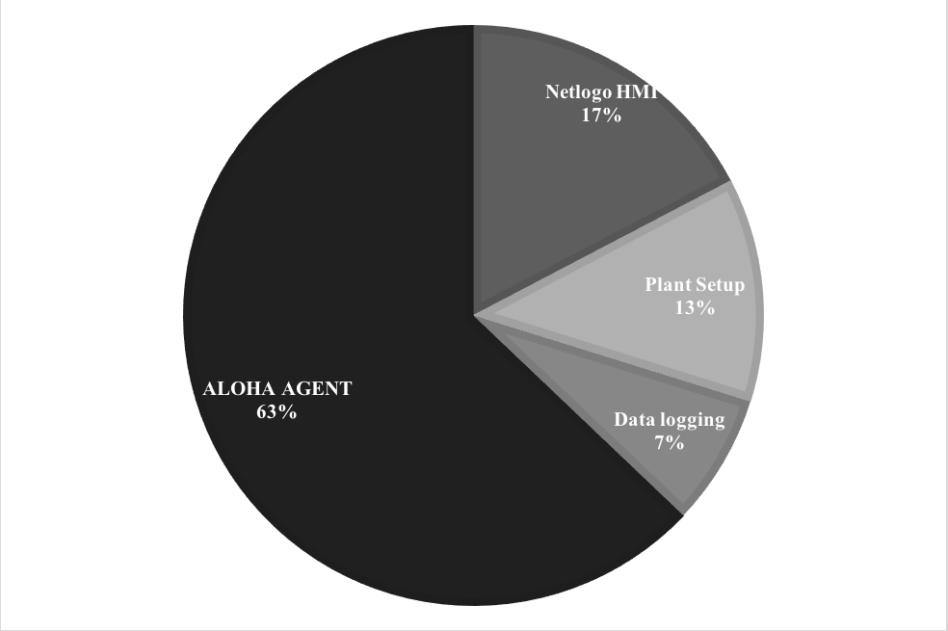


Fig. 29 Peso relativo del código para la aplicación con AGVs

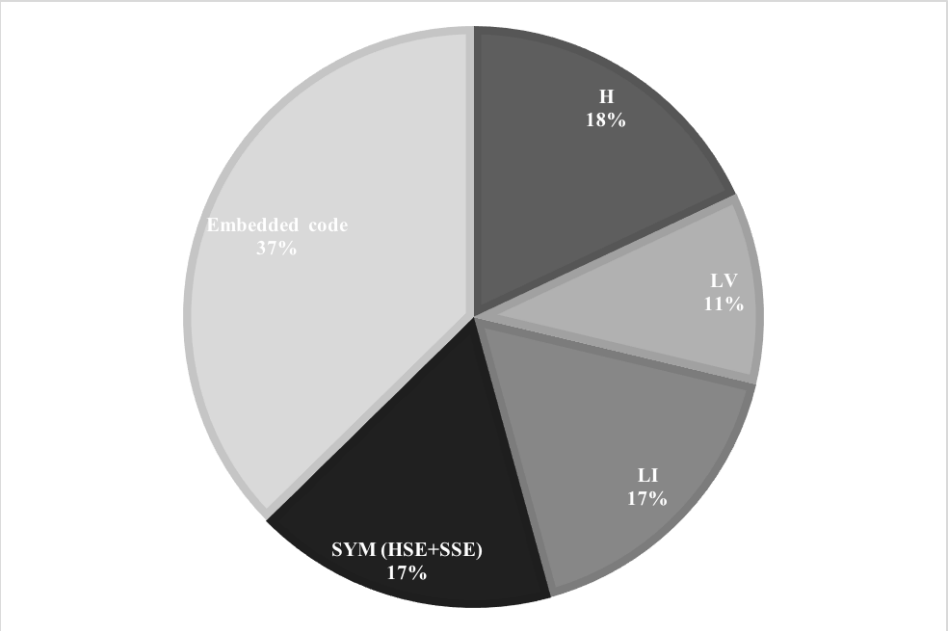


Fig. 30 Distribución del código en el agente ALOHA para aplicación con AGVs

A.2 Celda de producción

La aplicación de la celda de producción robotizada presenta una distribución relativa del código respecto al total del mismo tal como se muestra en la Fig. 31, siendo el código de mayor peso el que representa a los agentes ALOHA, ya que el mismo también incluye el código programado en el dispositivo embebido. A su vez, la Fig. 32 muestra cómo se distribuye el código fuente respecto a la arquitectura que presenta el agente ALOHA, donde se puede observar que el mayor porcentaje está representado por el código en el componente embebido.

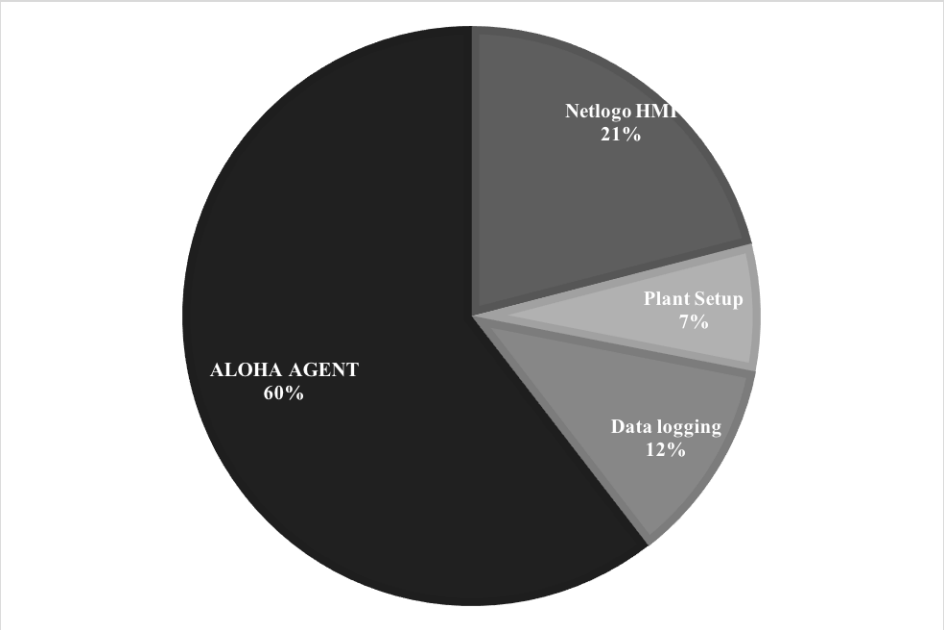


Fig. 31 Peso relativo del código para la aplicación con brazo robot

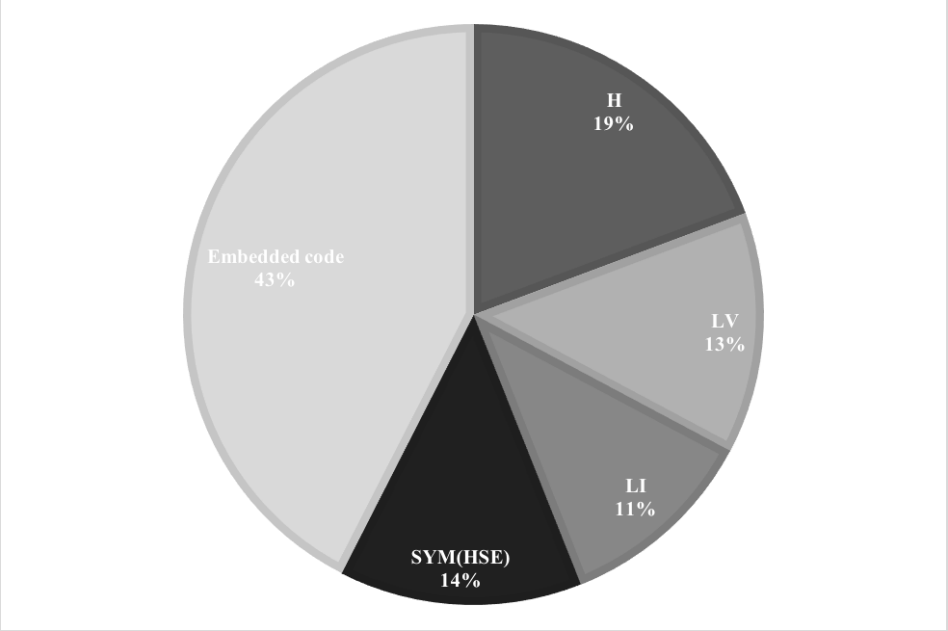


Fig. 32 Distribución del código en el agente ALOHA para aplicación con brazo robot

Bibliografía

[1] J. A. Sokolowski and C. M. Banks, Eds., *Modeling and simulation fundamentals: theoretical underpinnings and practical domains*. Hoboken, N.J: John Wiley, 2010.

[2] N. Metropolis and S. Ulam, "The Monte Carlo Method," *J. Am. Statical Assoc.*, vol. 44, no. 247, pp. 335–341, 1949.

[3] N. Metropolis, "The Beginning of the Monte Carlo Method," *Los Álamos*, vol. Special Issue, pp. 125–130, 1987.

[4] D. Goldsman, R. E. Nance, and J. R. Wilson, "A brief history of simulation revisited," in *Proceedings of the Winter Simulation Conference*, 2010, pp. 567–574.

[5] D. Mourtzis, M. Doukas, and D. Bernidaki, "Simulation in Manufacturing: Review and Challenges," *Procedia CIRP*, vol. 25, pp. 213–229, 2014.

[6] W. Halal, "The Information Technology Revolution Computer Hardware, Software and Services into the 21st Century," *Technol. Forecast. Soc. Change*, vol. 44, pp. 69–86, 1993.

[7] T. J. T. Bergin, "A history of the history of programming languages," *Commun. ACM*, vol. 50, no. 5, pp. 69–74, 2007.

[8] J. A. Botía, J. C. González, and J. Gomez y Juan Pavon, "The Ingenias Project: Methods And Tool For Developing Multiagent Systems," *Lat. Am. Trans. IEEE Rev. IEEE Am. Lat.*, vol. 6, no. 6, pp. 529–534, 2008.

[9] J. R. Thomson, "Chapter 8 - The Second Industrial Revolution – A Brief History of Computing," in *High Integrity Systems and Safety Management in Hazardous Industries*, J. R. Thomson, Ed. Boston: Butterworth-Heinemann, 2015, pp. 127 – 136.

[10] G. Clark, "Chapter 5 - The Industrial Revolution," in *Handbook of Economic Growth*, vol. 2, P. Aghion and S. N. Durlauf, Eds. Elsevier, 2014, pp. 217 – 262.

[11] T. Sauter, S. Soucek, W. Kastner, and D. Dietrich, "The Evolution of Factory and Building Automation," *IEEE Ind. Electron. Mag.*, vol. 5, no. 3, pp. 35–48, Sep. 2011.

[12] S. Weyer, M. Schmitt, M. Ohmer, and D. Gorecky, "Towards Industry 4.0 - Standardization as the crucial challenge for highly modular, multi-vendor production

systems.," presented at the International Federation of Automatic Control, 2015, vol. 48–3, pp. 579–584.

[13] C. Faller and D. Feldmüller, "Industry 4.0 Learning Factory for regional SMEs," *Procedia CIRP*, vol. 32, pp. 88–91, 2015.

[14] N. R. C. (US) B. on Manufacturing and E. D. C. on V. M. Challenges, *Visionary manufacturing challenges for 2020*. National Academies Press, 1998.

[15] R. Florida, "The New Industrial Revolution," *Butterworth-Heinemann Ltd*, no. FUTURES, pp. 559–576, 1991.

[16] B. Vogel-Heuser, A. Fay, I. Schaefer, and M. Tichy, "Evolution of software in automated production systems: Challenges and research directions," *J. Syst. Softw.*, vol. 110, pp. 54–84, Dec. 2015.

[17] R. Unland, "Chapter 2 - Industrial Agents," in *Industrial Agents*, P. L. Karnouskos, Ed. Boston: Morgan Kaufmann, 2015, pp. 23 – 44.

[18] A. W. Colombo, S. Karnouskos, J. M. Mendes, and P. Leitão, "Chapter 4 - Industrial Agents in the Era of Service-Oriented Architectures and Cloud-Based Industrial Infrastructures," in *Industrial Agents*, P. L. Karnouskos, Ed. Boston: Morgan Kaufmann, 2015, pp. 67 – 87.

[19] T. Strasser and A. Zoitl, "Chapter 5 - Distributed Real-Time Automation and Control - Reactive Control Layer for Industrial Agents," in *Industrial Agents*, P. L. Karnouskos, Ed. Boston: Morgan Kaufmann, 2015, pp. 89 – 107.

[20] B. Vogel-Heuser, P. Göhner, and A. Lüder, "Chapter 9 - Agent-Based Control of Production Systems—and Its Architectural Challenges," in *Industrial Agents*, P. L. Karnouskos, Ed. Boston: Morgan Kaufmann, 2015, pp. 153 – 170.

[21] L. Yu, A. Schüller, and U. Epple, "On the engineering design for systematic integration of agent-orientation in industrial automation," *ISA Trans.*, vol. 53, no. 5, pp. 1404–1409, Sep. 2014.

[22] M. Wooldridge, *An Introduction to MultiAgent Systems - Second Edition*. John Wiley & Sons, 2009.

[23] A. T. Al-Hammouri, "A comprehensive co-simulation platform for cyber-physical systems," *Comput. Commun.*, vol. 36, no. 1, pp. 8–19, Dec. 2012.

[24] Z. Zhang, E. Eyisi, X. Koutsoukos, J. Porter, G. Karsai, and J. Sztipanovits, "A co-simulation framework for design of time-triggered automotive cyber physical systems," *Simul. Model. Pract. Theory*, vol. 43, pp. 16–33, Apr. 2014.

[25] K. bin Hasnan, L. B. Saesar, and T. Herawan, "A Hardware-In-the-Loop Simulation and Test for Unmanned Ground Vehicle on Indoor Environment," *Procedia Eng.*, vol. 29, pp. 3904–3908, 2012.

[26] R. Chhabra and M. R. Emami, "A holistic concurrent design approach to robotics using hardware-in-the-loop simulation," *Mechatronics*, vol. 23, no. 3, pp. 335–345, Apr. 2013.

[27] M. Wetter, "Co-simulation of building energy and control systems with the Building Controls Virtual Test Bed," *J. Build. Perform. Simul.*, vol. 4, no. 3, pp. 185–203, Sep. 2011.

[28] W. Li, X. Zhang, and H. Li, "Co-simulation platforms for co-design of networked control systems: An overview," *Control Eng. Pract.*, vol. 23, pp. 44–56, Feb. 2014.

[29] C. CRESCENDO, "CRESCENDO," Seventh Framework Programme, 2011.

[30] P. Casoli, A. Gambarotta, N. Pompini, and L. Riccò, "Development and Application of Co-simulation and 'Control- oriented' Modeling in the Improvement of Performance and Energy Saving of Mobile Machinery," *Energy Procedia*, vol. 45, pp. 849–858, 2014.

[31] W. M. Loucks, B. J. Doray, and D. G. Agnew, "Experiences in Real-Time Hardware-Software Co-Simulation," Bell-Nothern Research Ltd., 1993.

[32] D. Bullock, B. Johnson, R. B. Wells, M. Kyte, and Z. Li, "Hardware-in-the-loop simulation," *Transp. Res. Part C Emerg. Technol.*, vol. 12, no. 1, pp. 73–89, Feb. 2004.

[33] R. Isermann, J. Schaffnit, and S. Sinsel, "Hardware-in-the-loop simulation for the design and testing of engine-control systems," *Control Eng. Pract.*, vol. 7, no. 5, pp. 643–653, 1999.

[34] L. Wang, Y. Zhang, C. Yin, H. Zhang, and C. Wang, "Hardware-in-the-loop simulation for the design and verification of the control system of a series–parallel hybrid electric city-bus," *Simul. Model. Pract. Theory*, vol. 25, pp. 148–162, Jun. 2012.

[35] B. Shirazi, I. Mahdavi, and N. Mahdavi-Amiri, "iCoSim-FMS: An intelligent co-simulator for the adaptive control of complex flexible manufacturing systems," *Simul. Model. Pract. Theory*, vol. 19, no. 7, pp. 1668–1688, Aug. 2011.

[36] *ICOS - Independent Co-Simulation*. Co-Simulation Research Group.

[37] L. J. De Vin, A. H. C. Ng, J. Oscarsson, and S. F. Andler, “Information fusion for simulation based decision support in manufacturing,” *Robot. Comput.-Integr. Manuf.*, vol. 22, no. 5–6, pp. 429–436, Oct. 2006.

[38] P. Novák, R. Šindelář, and R. Mordinyi, “Integration framework for simulations and SCADA systems,” *Simul. Model. Pract. Theory*, vol. 47, pp. 121–140, Sep. 2014.

[39] B. Rahmani and S. R. Hashemi, “Internet-based control of FCU hardware-in-the-loop simulators,” *Simul. Model. Pract. Theory*, vol. 56, pp. 69–81, Aug. 2015.

[40] M. Moallemi and G. Wainer, “Modeling and simulation-driven development of embedded real-time systems,” *Simul. Model. Pract. Theory*, vol. 38, pp. 115–131, Nov. 2013.

[41] I.-M. Chen, “Rapid response manufacturing through a rapidly reconfigurable robotic workcell,” *Robot. Comput.-Integr. Manuf.*, vol. 17, no. 3, pp. 199–213, 2001.

[42] M. Bruccoleri, “Reconfigurable control of robotized manufacturing cells,” *Robot. Comput.-Integr. Manuf.*, vol. 23, no. 1, pp. 94–106, Feb. 2007.

[43] C. Queiroz, A. Mahmood, and Z. Tari, “SCADASim—A Framework for Building SCADA Simulations,” *IEEE Trans. Smart Grid*, vol. 2, no. 4, pp. 589–597, Dec. 2011.

[44] H. Nylund and P. H. Andersson, “Simulation of service-oriented and distributed manufacturing systems,” *Robot. Comput.-Integr. Manuf.*, vol. 26, no. 6, pp. 622–628, Dec. 2010.

[45] M. Rolón and E. Martínez, “Agent-based modeling and simulation of an autonomic manufacturing execution system,” *Comput. Ind.*, vol. 63, no. 1, pp. 53–78, Jan. 2012.

[46] A. Baykasoğlu and V. Kaplanoğlu, “An application oriented multi-agent based approach to dynamic load/truck planning,” *Expert Syst. Appl.*, vol. 42, no. 15–16, pp. 6008–6025, Sep. 2015.

[47] D. Mourtzis, M. Doukas, and D. Bernidaki, “Simulation in Manufacturing: Review and Challenges,” *Procedia CIRP*, vol. 25, pp. 213–229, 2014.

[48] N. Ruiz, A. Giret, V. Botti, and V. Feria, “An intelligent simulation environment for manufacturing systems,” *Comput. Ind. Eng.*, vol. 76, pp. 148–168, Oct. 2014.

[49] L. Ribeiro, A. Rocha, A. Veiga, and J. Barata, “Collaborative routing of products using a self-organizing mechatronic agent framework—A simulation study,” *Comput. Ind.*, vol. 68, pp. 27–39, Apr. 2015.

[50] S. Rodriguez, V. Hilaire, and A. Koukam, “Towards a holonic multiple aspect analysis and modeling approach for complex systems: Application to the simulation of industrial plants,” *Simul. Model. Pract. Theory*, vol. 15, no. 5, pp. 521–543, May 2007.

[51] S. Rodriguez, “From Analysis to Design of Holonic Multi-Agent Systems: A Framework, Methodological Guidelines and Applications,” 2005.

[52] U. Manzoor and B. Zafar, “Multi-Agent Modeling Toolkit – MAMT,” *Simul. Model. Pract. Theory*, vol. 49, pp. 215–227, Dec. 2014.

[53] V. Mařík and J. Lažanský, “Industrial applications of agent technologies,” *Control Eng. Pract.*, vol. 15, no. 11, pp. 1364–1380, Nov. 2007.

[54] P. Novák, R. Šindelář, and R. Mordinyi, “Integration framework for simulations and SCADA systems,” *Simul. Model. Pract. Theory*, vol. 47, pp. 121–140, Sep. 2014.

[55] D. Böhnlein, K. Schweiger, and A. Tuma, “Multi-agent-based transport planning in the newspaper industry,” *Int. J. Prod. Econ.*, vol. 131, no. 1, pp. 146–157, May 2011.

[56] B. Shirazi, I. Mahdavi, and N. Mahdavi-Amiri, “iCoSim-FMS: An intelligent co-simulator for the adaptive control of complex flexible manufacturing systems,” *Simul. Model. Pract. Theory*, vol. 19, no. 7, pp. 1668–1688, Aug. 2011.

[57] V. Julián, “RT-MESSAGE: Desarrollo de Sistemas Multiagente de Tiempo Real,” 2002.

[58] “EURESCOM (2000). MESSAGE: Methodology for engineering systems of software agents. Initial methodology. Technical Report P907-D1, EU- RESCOM.”

[59] B. P. Douglass, *Doing Hard Time: Developing Real-time Systems with UML, Objects, Frameworks, and Patterns*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1999.

[60] V. Botti, C. Carrascosa, V. Julian, and J. Soler, “Modelling Agents in Hard Real-Time Environments,” in *Multi-Agent System Engineering*, vol. 1647, F. Garijo and M. Boman, Eds. Springer Berlin Heidelberg, 1999, pp. 63–76.

[61] V. Julián, M. Gonzalez, M. Rebollo, C. Carrascosa, and V. Botti, “InSiDE: una herramienta para el desarrollo de Agentes ARTIS,” in *Actas de SEID’2000*, 2000.

[62] G. Pedone, “Multi Agent Systems, Relating Artificial Intelligence Concepts and Information Technology Models in Competitive Agent-Based Applications,” 2011.

[63] J. Barata, “Coalition Based Approach for Shop Floor Agility,” 2003.

[64] “Protégé-2000,” Nov-2015. [Online]. Available: http://protege.stanford.edu.

[65] J. Barata and L. M. Camarinha-Matos, “Development of a FMS/FAS System - The CRI’s Pilot Unit, in Studies in Informatics and Control,” vol. 3(2–3), pp. 231–239, 1994.

[66] L. E. Henesey, “Multi-agent systems for container terminal management,” Blekinge Institute of Technology, Karlskrona, 2006.

[67] J. A. Botía, J. C. González, and J. Gomez y Juan Pavon, “The Ingenias Project: Methods And Tool For Developing Multiagent Systems,” *Lat. Am. Trans. IEEE Rev. IEEE Am. Lat.*, vol. 6, no. 6, pp. 529–534, 2008.

[68] J. J. Gómez-Sanz, C. R. Fernández, and J. Arroyo, “Model driven development and simulations with the INGENIAS agent framework,” *Simul. Model. Pract. Theory*, vol. 18, no. 10, pp. 1468–1482, Nov. 2010.

[69] “Egemin N.V. company and Distrinet research group, joint R&D, Architectural Design and Evaluation Of An Industrial AGV Transportation System With A Multiagent System Approach (SATURN 2006).” May-2006.

[70] M. Pěchouček and V. Mařík, “Industrial deployment of multi-agent technologies: review and selected case studies,” *Auton. Agents Multi-Agent Syst.*, vol. 17, no. 3, pp. 397–431, Dec. 2008.

[71] P. Farahvash and T. O. Boucher, “A multi-agent architecture for control of AGV systems,” *Robot. Comput.-Integr. Manuf.*, vol. 20, no. 6, pp. 473–483, Dec. 2004.

[72] S. C. Srivastava, A. K. Choudhary, S. Kumar, and M. K. Tiwari, “Development of an intelligent agent-based AGV controller for a flexible manufacturing system,” *Int. J. Adv. Manuf. Technol.*, vol. 36, no. 7–8, pp. 780–797, Mar. 2008.

[73] A. Wallace, “Multi-agent negotiation strategies utilizing heuristics for the flow of AGVs,” *Int. J. Prod. Res.*, vol. 45, no. 2, pp. 309–322, 2007.

[74] F. bin Md Fauadi, M. Hafidz, H. Lin, and T. Murata, “Dynamic task assignment of autonomous AGV system based on multi agent architecture,” in *Progress in Informatics and Computing (PIC), 2010 IEEE International Conference on*, 2010, vol. 2, pp. 1151–1156.

[75] R. Erol, C. Sahin, A. Baykasoglu, and V. Kaplanoglu, “A multi-agent based approach to dynamic scheduling of machines and automated guided vehicles in manufacturing systems,” *Appl. Soft Comput.*, vol. 12, no. 6, pp. 1720–1732, Jun. 2012.

[76] R. R. Barton and M. Meckesheimer, “Chapter 18 Metamodel-Based Simulation Optimization,” in *Simulation*, vol. 13, S. G. Henderson and B. L. Nelson, Eds. Elsevier, 2006, pp. 535 – 574.

[77] M. Mora, G. Forgionne, F. Cervantes, L. Garrido, J. N. D. Gupta, and O. Gelman, “Toward a Comprehensive Framework for the Design and Evaluation of Intelligent Decision-making Support Systems (i-DMSS),” *J. Decis. Syst.*, vol. 14, no. 3, pp. 321–344, 2005.

[78] A. Fernández-Caballero and J. Gascueña, “Developing Multi-Agent Systems through Integrating Prometheus, INGENIAS and ICARO-T,” in *Agents and Artificial Intelligence*, vol. 67, J. Filipe, A. Fred, and B. Sharp, Eds. Springer Berlin Heidelberg, 2010, pp. 219–232.

[79] H. Martínez-Barberá and D. Herrero-Pérez, “Autonomous navigation of an automated guided vehicle in industrial environments,” *Robot. Comput.-Integr. Manuf.*, vol. 26, no. 4, pp. 296–311, Aug. 2010.

[80] M. Short and K. Burn, “A generic controller architecture for intelligent robotic systems,” *Robot. Comput.-Integr. Manuf.*, vol. 27, no. 2, pp. 292–305, Apr. 2011.

[81] X. Desforges, A. Habbadi, and B. Archimède, “Design methodology for smart actuator services for machine tool and machining control and monitoring,” *Robot. Comput.-Integr. Manuf.*, vol. 27, no. 6, pp. 963–976, Dec. 2011.

[82] H. I. Bozma and M. E. Kalalıoğlu, “Multirobot coordination in pick-and-place tasks on a moving conveyor,” *Robot. Comput.-Integr. Manuf.*, vol. 28, no. 4, pp. 530–538, Aug. 2012.

[83] I.-M. Chen, “Rapid response manufacturing through a rapidly reconfigurable robotic workcell,” *Robot. Comput.-Integr. Manuf.*, vol. 17, no. 3, pp. 199–213, 2001.

[84] Z. Pan, J. Polden, N. Larkin, S. Van Duin, and J. Norrish, “Recent progress on programming methods for industrial robots,” *Robot. Comput.-Integr. Manuf.*, vol. 28, no. 2, pp. 87–94, Apr. 2012.

[85] Z. M. Bi, Y. Lin, and W. J. Zhang, “The general architecture of adaptive robotic systems for manufacturing applications,” *Robot. Comput.-Integr. Manuf.*, vol. 26, no. 5, pp. 461–470, Oct. 2010.

[86] M. Bruccoleri, “Reconfigurable control of robotized manufacturing cells,” *Robot. Comput.-Integr. Manuf.*, vol. 23, no. 1, pp. 94–106, Feb. 2007.

[87] K. Alexopoulos, D. Mavrikios, and G. Chryssolouris, Eds., *ErgoToolkit: an ergonomic analysis tool in a virtual manufacturing environment*, vol. 26. London [u.a.]: Taylor & Francis, 2013.

[88] T. Holz, A. G. Campbell, G. M. P. O’Hare, J. W. Stafford, A. Martin, and M. Dragone, “MiRA—Mixed Reality Agents,” *Int. J. Hum.-Comput. Stud.*, vol. 69, no. 4, pp. 251–268, Apr. 2011.

[89] “Industrie 4.0 - Innovationen für die Produktion von morgen,.” Bundesministerium für Bildung und Forschung (BMBF), Apr-2015.

[90] S. Weyer, M. Schmitt, M. Ohmer, and D. Gorecky, “Towards Industry 4.0 - Standardization as the crucial challenge for highly modular, multi-vendor production systems.,” presented at the IFAC-PapersOnLine, 2015, vol. 48–3, pp. 579–584.

[91] D. Kolberg and D. Zühlke, “Lean Automation enabled by Industry 4.0 Technologies,” presented at the IFAC-PapersOnLine, 2015, vol. 48–3, pp. 1870–1875.

[92] C. Faller and D. Feldmüller, “Industry 4.0 Learning Factory for regional SMEs,” *Procedia CIRP*, vol. 32, pp. 88–91, 2015.

[93] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*. Englewood Cliffs, N.J: Prentice Hall, 1995.

[94] M. Wooldridge and N. R. Jennings, “Intelligent agents - Theory and Practice,” *Knowl. Eng. Rev.*, vol. 10(2), pp. 115 – 152, 1995.

[95] R. Allan, “Survey of Agent Based Modelling and Simulation Tools,” Computational Science and Engineering Department, STFC Daresbury Laboratory, Daresbury, UK, Jun. 2009.

[96] “OpenABM, Modeling Platforms,” Dec-2015. [Online]. Available: https://www.openabm.org/page/modeling-platforms.

[97] Wikipedia, “Comparison of agent-based modeling software,” Dec-2015. [Online]. Available: https://en.wikipedia.org/wiki/Comparison_of_agent-based_modeling_software.

[98] A. Koestler, *The Ghost in the Machine*. Hutchinson, 1967.

[99] H. Van Brussel, J. Wyns, P. Valckenaers, L. Bongaerts, and P. Peeters, “Reference Architecture for Holonic Manufacturing Systems: PROSA,” *Comput Ind*, vol. 37, no. 3, pp. 255–274, Nov. 1998.

[100] N. R. Jennings, “Coordination Techniques for Distributed Artificial Intelligence,” in *Foundations of Distributed Artificial Intelligence*, G. M. P. O’Hare and N. R. Jennings, Eds. New York, NY, USA: John Wiley & Sons, Inc., 1996, pp. 187–210.

[101] P. Iñigo-Blasco, F. Diaz-del-Rio, M. C. Romero-Ternero, D. Cagigas-Muñiz, and S. Vicente-Diaz, “Robotics software frameworks for multi-agent robotic systems development,” *Robot. Auton. Syst.*, vol. 60, no. 6, pp. 803–821, Jun. 2012.

[102] B. Jerry, *Discrete-event system simulation*. Pearson Education India, 1984.

[103] Wikipedia, “Simulation,” Dec-2015. [Online]. Available: https://en.wikipedia.org/wiki/Simulation.

[104] S. F. Railsback, S. L. Lytinen, and S. K. Jackson, “Agent-based Simulation Platforms: Review and Development Recommendations,” *SIMULATION*, vol. 82, no. 9, pp. 609–623, Sep. 2006.

[105] AgentLink, “Agent Software,” Dec-2015. [Online]. Available: http://eprints.agentlink.org/view/type/software.html.

[106] I. F. Chaile and Ribas-Xirgo, “A development environment for indoor multiagent transport system with miniAGV,” presented at the Workshop of Physical Agents (WAF), Santiago de Compostela, Spain, 2012.

[107] J. A. Sokolowski and C. M. Banks, Eds., *Modeling and simulation fundamentals: theoretical underpinnings and practical domains*. Hoboken, N.J: John Wiley, 2010.

[108] J. A. Sokolowski and C. M. Banks, *Principles of modeling and simulation: A multidisciplinary approach*. John Wiley & Sons, 2011.

[109] N. Ruiz, “Simulación Asistida por Agentes para Sistemas de Fabricación Inteligentes,” 2009.

[110] C. A. Chung, *Simulation modeling handbook: a practical approach*. Boca Raton: CRC Press, 2004.

[111] C. Ptolemaeus, Ed., *System design, modeling, and simulation: using Ptolemy II*, 1. ed., version 1.02. Berkeley, Calif: UC Berkeley EECS Dept, 2014.

[112] C. J. Castle and A. T. Crooks, "Principles and concepts of agent-based modelling for developing geospatial simulations," 2006.

[113] "NetLogo itself: Wilensky, U. 1999. NetLogo. http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling, Northwestern University. Evanston, IL." .

[114] R.-S. Chen, K.-Y. Lu, and C. C. Chang, "Intelligent warehousing management systems using multi-agent," *Int J Comput Appl Technol*, vol. 4, pp. 194–201, 2003.

[115] E. Di Lello and A. Saffiotti, "The PEIS Table An Autonomous Robotic Table for Domestic Environments," *Autom. Control Meas. Electron. Comput. Commun.*, vol. 52, no. 3, pp. 244–255, 2011.

[116] E. Ivanjko, I. Petrović, and M. Vašak, "Sonar-based Pose Tracking of Indoor Mobile Robots," *Autom. Control Meas. Electron. Comput. Commun.*, vol. 45, no. 3–4, pp. 145–154, 2004.

[117] F. Mastrogiovanni, A. Scalmato, A. Sgorbissa, and R. Zaccaria, "Robots and Intelligent Environments: Knowledge Representation and Distributed Context Assessment," *Autom. Control Meas. Electron. Comput. Commun.*, vol. 52, no. 3, 2011.

[118] M. Al khawaldah and A. Nuechter, "Multi-Robot Cooperation for Efficient Exploration," *Autom. – J. Control Meas. Electron. Comput. Commun.*, vol. 55, no. 3, Dec. 2014.

[119] G. Klančar, M. Brezak, D. Matko, and I. Petrović, "Mobile robots tracking using computer vision," *Autom. Control Meas. Electron. Comput. Commun.*, vol. 46, no. 3–4, pp. 155–163, 2006.

[120] J. Stückler and S. Behnke, "Dynamaid, an Anthropomorphic Robot for Research on Domestic Service Applications," *Autom. Control Meas. Electron. Comput. Commun.*, vol. 52, pp. 233–243, 2011.

[121] "Reina Sofía Análisis Clínicos," Dec-2009. [Online]. Available: https://www.youtube.com/watch?v=_u64EUrFF4I.

[122] L. Ribas-Xirgo and I. F. Chaile, “An Agent-based Model of Autonomous Automated-Guided Vehicles for Internal Transportation in Automated Laboratories.,” in *ICAART (1)*, 2013, pp. 262–268.

[123] L. Ribas-Xirgo, J. M. Moreno-Villafranca, and I. F. Chaile, “On using automated guided vehicles instead of conveyors,” in *Emerging Technologies & Factory Automation (ETFA), 2013 IEEE 18th Conference on*, 2013, pp. 1–4.

[124] J. Himoff, G. Rzevski, and P. Skobelev, “Magenta technology multi-agent logistics i-Scheduler for road transportation,” in *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, 2006, pp. 1514–1521.

[125] “Roche Cobas 8000 analyzer,” 22-Dec-2014. [Online]. Available: https://www.youtube.com/watch?v=lJaTt_S7zW8.

[126] L. Ribas-Xirgo and I. F. Chaile, “Agent-based Automatic Guided Vehicle (AGV) System Development Framework,” Oct-2015. [Online]. Available: https://www.youtube.com/watch?v=pJmNw24aBdQ.

[127] Roche, “Cobas 8000 modular analyzer series brochure,” Oct. 2015.

[128] I. F. Chaile and L. Ribas-Xirgo, “Controlling multiple AGVs with agent-based-modelling,” Oct-2015. [Online]. Available: https://www.youtube.com/watch?v=2LdK5AhJhuQ.

[129] I. F. Chaile and L. Ribas-Xirgo, “Synchronizing automated guided vehicles (AGV) with agent-based model (ABM) for control,” Oct-2015. [Online]. Available: https://www.youtube.com/watch?v=A4tk7kRtjiA.

[130] “FIPA ACL Message Structure Specification,” Sep. 2015.

[131] J. Rinaldi, *OPC UA: The Basics: An OPC UA Overview For Those Who May Not Have a Degree in Embedded Programming*. CreateSpace, 2013.

[132] A. Zoitl, *Real-time Execution for IEC 61499*. ISA, 2008.

[133] R. Zurawski, Ed., “Section 6 – Real-Time Embedded Systems, In The Industrial Information Technology Handbook,” CRC Press LLC, 2005.

[134] R. Williams, Ed., “Chapter 1 – Introduction to real-time systems,” in *Real-Time Systems Development*, Oxford: Butterworth-Heinemann, 2006, pp. 1–28.

[135] R. Williams, “Chapter 2 – Implementing simple real-time systems,” in *Real-Time Systems Development*, Oxford: Butterworth-Heinemann, 2006, pp. 29–45.

[136] P. Zhang, “Chapter 16 – Real-time operating systems,” in *Advanced Industrial Control Technology*, Elsevier, 2010, pp. 613–683.

[137] P. Caspi and O. Maler, “From Control Loops to Real-Time Programs,” Sep-2015. [Online]. Available: http://www-verimag.imag.fr/~maler/Papers/caspimaler.pdf, .

[138] L. Ribas-Xirgo, A. Miro-Vicente, I. F. Chaile, and A. J. Velasco-González, “Multi-agent model of a sample transport system for modular in-vitro diagnostics laboratories,” in *Emerging Technologies & Factory Automation (ETFA), 2012 IEEE 17th Conference on*, 2012, pp. 1–8.

[139] L. Ribas-Xirgo and I. F. Chaile, “Multi-agent-based controller architecture for AGV systems,” in *Emerging Technologies & Factory Automation (ETFA), 2013 IEEE 18th Conference on*, 2013, pp. 1–4.

[140] I. F. Chaile and L. Ribas-Xirgo, “MASYM, a Framework to Deploy Synchronized Industrial Systems Based on Any ABM Simulator,” *IEEE IEEE Lat. Am. Trans.*, Oct. 2015.

[141] I. F. Chaile and L. Ribas-Xirgo, “How to rapidly deploy scale prototypes of transportation systems based on robots,” *InTech Int. J. Adv. Robot. Syst.*, Under minor review 2015.

[142] I. F. Chaile and L. Ribas-Xirgo, “Running ABM simulations synchronized with reality to control transport systems,” *KoREMA, Automatika – Journal for Control, Measurement, Electronics, Computing and Communications*, Under minor review-2015.

[143] I. F. Chaile and L. Ribas-Xirgo, “Agent simulator-based control architecture for rapid development of multi-robot systems,” presented at the International Conference on Systems, Control, Signal Processing and Informatics (SCSI), INASE Joint Conferences, 2015, pp. 126–134.