risti
Revista Ibérica de Sistemas e Tecnologias de Informação
Revista Ibérica de Sistemas y Tecnologías de Información

Recebido/Submission: 10/01/2020
Aceitação/Acceptance: 15/04/2020

Nuevos Aportes de las Tecnologías de Información para el Desarrollo de Simulación Distribuida

Juan Leonardo Sarli[1], María Julia Blas[1], Silvio Gonnet[1]

{ juanleonardosarli, mariajuliablas, sgonnet }@santafe-conicet.gov.ar

[1] INGAR – Instituto de Desarrollo y Diseño, CONICET-UTN, Santa Fe, CP: 3000, Santa Fe, Argentina.

DOI: 10.17013/risti.37.65–81

Resumen: Se entiende por simulación al proceso por medio del cual se representa, reproduce o imita el comportamiento observable de un proceso o sistema real a lo largo del tiempo y el espacio. La simulación distribuida tiene la capacidad de acelerar la ejecución de un único modelo, vincular y reutilizar múltiples modelos para simular modelos más grandes y acelerar la ejecución de etapas de experimentación. En este contexto, la construcción de simulaciones distribuidas ha mejorado en los últimos años gracias al surgimiento de nuevas tecnologías de la información. En este artículo se describen los principios, modos de trabajo y enfoques de administración de tiempo asociados a esta técnica junto con las herramientas de software que, en la actualidad, brindan soporte a su aplicación. Además, se presenta una revisión bibliográfica que evidencia el crecimiento (y la importancia) de su uso como técnica de estudio en diferentes dominios.

Palabras-clave: computación distribuida y paralela; gestión del tiempo.

New Contributions of Information Technologies to Develop Distributed Simulation

Abstract: Simulation is the process by which the observable behavior of a real process or system is represented, reproduced or imitated in time and space. Distributed simulation can be used for accelerate the execution of models, reuse models in larger models, and accelerate the execution of experiments. Given the emergence of new information technologies, the use of distributed simulation has grown. This paper describes the fundamentals, modes and time management approaches used in distributed simulations along with the software tools that improves its development. Also, a literature review is presented to show how this technique is applied in distinct domains.

Keywords: parallel and distribute computing; time management.

1. Introducción

La simulación es una técnica que permite describir y analizar el comportamiento de un sistema real, en virtud de detectar comportamientos emergentes y/o utilizar las

salidas de información para apoyar el diseño y mejora de dicho sistema. En general, esta técnica es utilizada cuando: *i)* resulta muy difícil (o incluso imposible) desarrollar un modelo analítico para representar el sistema, *ii)* el sistema posee variables aleatorias, *iii)* la dinámica del sistema es compleja y produce comportamientos emergentes, o *iv)* el objetivo es observar el comportamiento del sistema en un período de tiempo dado (Powers, Sanchez, & Lucas, 2012; Rainey & Tolk, 2015; Anagnostou & Taylor, 2017). En cualquier caso, el uso de técnicas de simulación provee una herramienta de soporte a la toma de decisiones tanto a nivel de diseño (es decir, en una etapa previa a la construcción del sistema) como a nivel operativo (por ejemplo, probando diferentes políticas de funcionamiento antes de su aplicación sobre el sistema real). Luego, la simulación ayuda a identificar problemas relevantes, evaluar cuantitativamente soluciones alternativas y detectar oportunidades de mejora.

En este contexto, la *simulación distribuida* ha surgido como un campo híbrido que combina raíces de la computación paralela con la computación distribuida. La *simulación paralela* surge con el objetivo de incrementar la velocidad de ejecución de una *simulación*[1] por medio del uso de múltiples procesadores en una única computadora. Por su parte, en sus orígenes, la *simulación distribuida* investigaba la forma en la cual interconectar múltiples simulaciones (ejecutadas en distintas computadoras distribuidas geográficamente y conectadas por medio de una red) en una única simulación (Fujimoto, 2016). Sin embargo, dado el surgimiento de los enfoques basados en *grid computing* y *cloud computing*, en la actualidad las diferencias entre ambos tipos de simulación son cada vez menos claras (Fujimoto, 2016). Por este motivo, los autores han comenzado a utilizar indistintamente el término *simulación distribuida* como denominador común de ambos casos.

Recientemente se ha observado un incremento en el uso de este tipo de simulación como mecanismo de soporte al estudio en diferentes sistemas. Dado que la simulación distribuida tiene la capacidad de acelerar la ejecución de un único modelo, vincular y reutilizar múltiples modelos para simular modelos más grandes y acelerar la etapa de experimentación; los diversos beneficios de esta técnica son apropiados para el estudio de grandes sistemas. En este sentido, el surgimiento de herramientas de software específicas para su construcción ha contribuido al éxito de su aplicación en diferentes ámbitos como ser, por ejemplo, el área de defensa, diseño de sistemas, cadenas de suministro, elaboración de productos y construcción de ciudades inteligentes.

Este trabajo introduce las principales características de la simulación distribuida, brindando una descripción de los enfoques existentes y las tecnologías disponibles para su aplicación. Además, presenta una revisión de la literatura en virtud de analizar las áreas de aplicación recientemente abordadas a nivel académico y la forma en la cual los enfoques y herramientas existentes brindan soporte cada una de ellas. De esta manera, se analiza el uso de la simulación distribuida en desarrollos relacionados con distintas áreas, brindado detalles de los casos de éxito junto con las limitaciones que restringen su uso en otro tipo de contextos.

[1] Refiere al par *(modelo, simulador)* donde el *modelo* es el comportamiento a ejecutar y el *simulador* es la herramienta en la que se ejecuta dicho comportamiento.

La Sección 2 introduce los principios de la simulación distribuida en base a los modos de operación y enfoques de manejo de tiempo. La Sección 3 presenta las tecnologías existentes para la construcción de este tipo de esquemas de simulación, describiendo un conjunto de herramientas de software disponibles. La Sección 4 analiza el uso de simulación distribuida en nueve áreas (defensa, medio ambiente, energía, arquitectura de computadoras y sistemas de control, salud, manufactura, redes, espacio y transporte), las cuales se obtienen a partir de una revisión de la literatura existente. Incluye además una breve reseña de las potenciales oportunidades que presenta esta técnica en problemas vigentes. Finalmente, la Sección 5 se encuentra destinada a las conclusiones.

2. Principios de la Simulación Distribuida

La simulación distribuida posee múltiples ventajas en comparación con la simulación local. Sin embargo, el cambio de paradigma de un contexto local a un entorno distribuido conlleva a la necesidad de resolver dos problemas fundamentales: la forma en la cual se genera la distribución de las simulaciones (*modo de operación*) y el enfoque que se utiliza para coordinar la comunicación entre simulaciones (*manejo del tiempo*).

2.1. Modo de Operación

Los modos de operación varían según el objetivo final de la simulación. En términos generales, estos objetivos pueden definirse como: *i)* uso de técnicas de computación paralela y distribuida para incrementar la velocidad en la ejecución de un programa de simulación y/o para integrar diversas simulaciones en un único programa -a efectos de fomentar el reúso de simulaciones- (Fujimoto, 2016; Taylor, 2019); *ii)* realización de experimentos de forma paralela a fin de reducir el tiempo invertido en el análisis del sistema (Yücesan, Luo, Chen, & Lee, 2001).

Según estos objetivos, Taylor (2019) ha definido tres modos de operación básicos para la simulación distribuida, a saber: *i) modo A* para acelerar una única simulación, *ii) modo B* para interconectar y reutilizar distintas simulaciones, y *iii) modo C* para acelerar la experimentación sobre una única simulación. La Figura 1 esquematiza los tres modos de operación diferenciando, para cada caso, el esquema de simulación local contra el esquema de simulación distribuida.

El *modo de operación A* (Figura 1a) plantea una simulación global dividida en diversos submodelos que, durante su ejecución, requieren intercambiar información. En el esquema de simulación tradicional (local), los submodelos son ejecutados en una única computadora por lo que el intercambio de información se da de forma directa. En este caso el tiempo de simulación total es la suma del tiempo de ejecución de los submodelos. En cambio, en el esquema de simulación distribuida los submodelos se ejecutan en distintas computadoras e intercambian información por medio de una red de comunicaciones. Luego, la ejecución de submodelos en paralelo da lugar a ejecuciones más rápidas que la simulación tradicional, reduciendo el tiempo total asociado al modelo global.

El *modo de operación B* (Figura 1b) centra su atención en múltiples modelos de simulación (independientes, por lo que no requieren intercambiar información) que son ejecutados en distintas computadoras. En este caso, una configuración de simulación distribuida posibilita la construcción de un nuevo modelo de simulación de mayor tamaño (tanto en

lo referido a la cantidad de modelos como así también a su complejidad) cuya ejecución va más allá de la capacidad de una única computadora. Para esto, el esquema plantea vínculos entre los distintos modelos a fin de conformar una única simulación por medio de su interacción en base a una red de comunicaciones. Adicionalmente, el reúso de modelos de simulación (dado por el establecimiento de interconexiones basadas en red), reduce potencialmente el costo de desarrollar nuevas simulaciones.

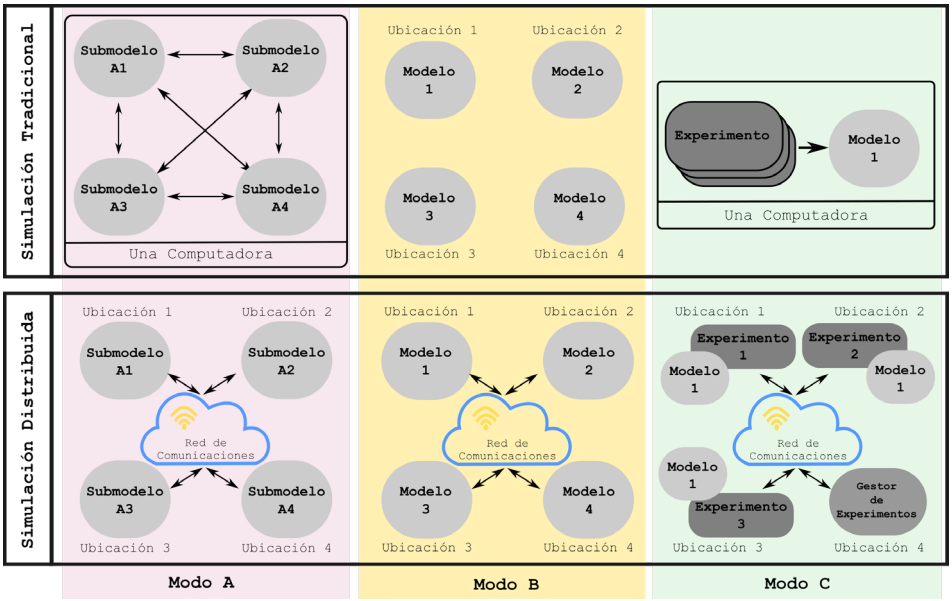

Figura 1 – Esquematización de los modos de operación en simulación distribuida.

Finalmente, en el *modo de operación C* (Figura 1c) la base de trabajo es la realización de múltiples experimentos sobre un mismo modelo de simulación. Para el caso de simulación tradicional, los experimentos se realizan de forma secuencial (es decir, uno a la vez) sobre un único modelo de simulación ubicado en una misma computadora. Por el otro lado, en el caso de la simulación distribuida se observa la ubicación de réplicas del modelo de simulación original en diversas computadoras. Los diferentes experimentos se distribuyen sobre las máquinas, en tanto que la coordinación de ejecuciones y resultados se da a través de un gestor de experimentos (el cual se comunica con las computadoras por medio de una red de comunicaciones). De esta forma se paraleliza la ejecución de experimentos, lo que da como consecuencia una disminución en los tiempos de experimentación o un eventual incremento del número de experimentos en un mismo período de tiempo.

2.2. Manejo del Tiempo

El problema de administración del tiempo refiere a la forma en la cual se sincronizan/ coordinan las simulaciones que participan en una simulación distribuida, con el objetivo

de garantizar que los resultados producidos serán iguales a los que se darían si dicha simulación fuese ejecutada de forma local.

Los simuladores que componen una simulación distribuida no intercambian eventos de forma continua. En este sentido, uno de los objetivos de este tipo de simulación es sacar ventaja de dicha situación a fin de asegurar que, al finalizar la ejecución, cada simulador componente ha procesado todos sus eventos (tanto internos como externos) de forma ordenada según su marca temporal. Esta característica se denomina *restricción de causalidad local*. Cuando todos los simuladores que componen una simulación distribuida cumplen con esta restricción, se dice que la simulación es causalmente correcta (Fujimoto, 2016). Adicionalmente, con una simulación distribuida se busca maximizar la cantidad de simulaciones que corren en paralelo a fin de reducir la sincronización requerida (lo que permite aprovechar el poder de procesamiento de las múltiples computadoras). Según como se combinan estos objetivos y cuál sea su prioridad, existen tres enfoques de administración del tiempo: *conservador*, *optimista* y *tiempo real*.

El *enfoque conservador* busca generar simulaciones causalmente correctas. El principal problema a resolver en este caso, es la forma en la cual un simulador determina que puede ejecutar un evento inminente sin generar problemas temporales (es decir, sin violar la restricción de causalidad). El objetivo es que el simulador garantice que no existe (ni existirá) otro evento cuya marca temporal sea anterior a la del evento que esta por ejecutar (evento inminente).

Una forma de solucionar este problema es que cada simulador posea una cola de eventos de entrada por cada *simulador influyente* (es decir, cada simulador del que puede recibir eventos). Si cada cola es ordenada por marca temporal (en sentido creciente), la selección del próximo evento a ejecutar simplemente consistirá en verificar el primer evento de cada cola y seleccionar el que tenga menor marca temporal. Sin embargo, este esquema no es útil cuando un simulador posee más de un simulador influyente ya que pueden presentarse *bloqueos mortales*.

Un *bloqueo mortal* se presenta cuando un simulador no procesa eventos de su cola de entrada debido a que se encuentra esperando (indefinidamente) un evento de un simulador influyente. Por ejemplo, sea A un simulador influenciado por los simuladores B y C. Asumiendo que A recibe dos eventos $e1$ y $e2$ provenientes de B y C, respectivamente, cuyas marcas temporales son $t1$ y $t2$. Si $t1 < t2$, el simulador A procesará $e1$. En caso contrario, procesará $e2$. Sin embargo, una vez elegido el primer evento inminente, el simulador A deberá esperar hasta recibir el próximo evento del simulador influyente ya procesado, a fin de garantizar que este nuevo evento no se encuentra planificado antes del que ya se encuentra en cola. Es decir, suponiendo que $t1 < t2$, el simulador A procesará $e1$. Sin embargo, antes de tomar la decisión de procesar $e2$, el simulador A debe esperar un nuevo evento de B (denominado $e3$, con marca temporal $t3$) que le garantice que $t2 < t3$. Si B no emite nuevos eventos hacia A, entonces los simuladores entran en un *bloqueo mortal*.

Para garantizar que no existan bloqueos mortales, el *enfoque conservador* utiliza mensajes nulos. Cuando un simulador recibe un mensaje nulo, se encuentra habilitado para continuar procesando los eventos ya presentes en su cola de entrada. Para que la

emisión de mensajes nulos evite bloqueos mortales, es necesario calcular el *lookahead*. El *lookahead* es el mínimo período de tiempo que un simulador puede garantizar la inexistencia de eventos, a partir del último envío y hasta el próximo envío. En otras palabras, es el tiempo durante el cual un simulador no generará salidas, medido desde el último evento enviado.

Para el cálculo del *lookahead* se estima un tiempo mínimo para el próximo evento de salida, el cual puede variar según se asuma que se reciben o no eventos externos. En su versión más simple, el valor del *lookahead* representa la demora de propagación mínima, para que un evento llegue desde el simulador influyente hacia el simulador destino. Sin embargo, el rendimiento de un *esquema conservador* depende (en gran medida) del valor del *lookahead*, a saber: *i)* valores pequeños conllevan una sobrecarga en el intercambio de mensajes y uso de procesamiento para mensajes nulos, *ii)* valores muy grandes disminuyen la sobrecarga de mensajes (procesando menor cantidad de mensajes nulos), pero pueden generar conflictos de estimación. En este contexto, se han llevado a cabo investigaciones con el objetivo de mejorar los valores de *lookahead*, a fin de mejorar el rendimiento en el *enfoque conservador* (Zacharewicz, Frydman, & Giambiasi, 2006).

A diferencia del *enfoque conservador*, en un *enfoque optimista* se procesan los eventos sin validar la condición de causalidad. Por este motivo pueden darse problemas de causalidad, lo que da como resultado que los simuladores deban recuperarse de dicha situación, a fin de garantizar la causalidad local de sus eventos. El objetivo de este enfoque es maximizar el paralelismo para ejecutar la simulación en menor tiempo.

Uno de los algoritmos optimistas más difundidos es *Time Warp*. En este algoritmo cada simulador maneja tres colas: *mensajes de entrada*, *mensajes de salida* y *estado del simulador*. Cuando un simulador recibe un mensaje, el mismo es enviado a la cola de *mensajes de entrada*. Para cada mensaje recibido, se procesa el evento y se produce un nuevo estado en tiempo *t*. Este estado es almacenado en la *cola de estados del simulador*. Luego, el simulador envía mensajes a los destinos requeridos y almacena en su cola de *mensajes de salida* los eventos enviados. Si un simulador recibe un mensaje con tiempo menor al tiempo actual, este evento está desordenado (*evento straggle*). Esto provoca una violación a la causalidad y, por lo tanto, debe ser corregida. En estos casos, el simulador debe ejecutar un *rollback*.

Realizar un *rollback* implica que el simulador debe: *i)* retornar al estado previo al tiempo del *evento straggle*, y *ii)* deshacer todos los eventos de mensajes enviados con tiempo posterior al nuevo estado actual. Para deshacer los eventos enviados, el simulador verifica la *cola de mensajes de salida* y envía una copia de los mensajes a deshacer con valores negativos (*antimensajes*). Cuando un simulador recibe un *antimensaje*, debe realizar su propio *rollback* para volver a un estado que respete la causalidad. Una vez que todos los *antimensajes* son enviados, el simulador que se encuentra ejecutando el *rollback* procesa los eventos correctamente ordenados en su cola de *mensajes de entrada* (por lo que continúa funcionando tal como lo estaba haciendo antes de la ocurrencia del evento *straggle*).

Uno de los mayores inconvenientes que posee el *enfoque optimista* es la necesidad de almacenamiento y rápido acceso a un gran número de estados por cada simulador.

Además, la ocurrencia de múltiples *rollbacks* en cascada puede dar lugar a problemas de rendimiento y demoras excesivas. Una alternativa para solucionar estos problemas es aplicar el mecanismo de *Global Virtual Time* (GVT). Este mecanismo establece un límite inferior sobre el tiempo global en el cual no se recibirán eventos desordenados. De esta manera, asegura que un simulador no realizará un *rollback* en un tiempo previo a dicho límite (por lo que se puede eliminar toda la información relacionada a los estados y colas de mensajes asociados a un tiempo previo al límite definido). Investigaciones recientes presentan mejoras para disminuir el número de *rollbacks* en cascada (y así no reprocesar eventos que no modifican el estado del simulador) realizando un cálculo eficiente del límite inferior del mecanismo GVT (Venu & Joe, 2014).

Tanto el *enfoque conservador* como el *enfoque optimista* soportan el análisis de sistemas mediante técnicas de simulación de eventos discretos. Sin embargo, en el *enfoque en tiempo real* los simuladores intentan responder de forma similar al mundo real o, al menos, entregar una respuesta en tiempo real a las acciones de los usuarios. Luego, en una simulación distribuida en tiempo real, cada simulador interactúa con múltiples simuladores brindando su respuesta a las interacciones en un período de tiempo lo más cercano posible al tiempo real.

En este enfoque de administración del tiempo, los diferentes simuladores intercambian información de estado por medio de un componente especial denominado *coordinador*. El *coordinador* recibe la información que produce cada simulador y, al mismo tiempo, es el encargado de retransmitir esa información hacia cada uno de los simuladores interesados. Para determinar cuáles son los simuladores interesados en enviar o recibir cierta información, el *coordinador* suele utilizar mecanismos *publish/subscribe*. Este tipo de mecanismos permite que cada simulador publique en el *coordinador* la información que es capaz de generar y, al mismo tiempo, se suscriba al *coordinador* indicando la información que espera recibir. Luego, para llevar a cabo la simulación global, los simuladores participantes consultan al coordinador acerca de la existencia de eventos de su interés (en una etapa previa a la ejecución de su próximo ciclo de simulación).

Para cumplir con este objetivo, se requiere de diferentes enfoques que posibiliten balancear el procesamiento y la comunicación entre los distintos simuladores. En este contexto, se han especificado diversos estándares en virtud de definir *protocolos de comunicación* (mensajes a usar para la comunicación entre simuladores), *formato de información a ser transmitida* (tipo de información que se intercambia a nivel simulador) y *software de soporte a la tecnología*. Uno de los estándares más difundidos es *High Level Architecture* (HLA) (IEEE, 2010).

3. Tecnologías para el Desarrolo de Simulaciones Distribuidas

En los últimos años se han realizado numerosos esfuerzos para el desarrollo de librerías de software que faciliten la construcción de simulaciones distribuidas. En su forma más simple, el desarrollo de simulaciones distribuidas se formuló en base a *lenguajes de simulación*. En este caso, uno de los primeros ejemplos fue el lenguaje de simulación de eventos discretos Maisie seguido por el lenguaje APOSTLE. Mientras

que el primero soporta tanto la simulación tradicional como la distribuida con enfoque conservador u optimista; el segundo permite únicamente el desarrollo de simulaciones distribuidas con administración del tiempo optimista. Recientemente, Falcone et al. (2017) han presentado un kit de desarrollo junto con una metodología para su uso en base a anotaciones Java. Este kit posibilita la reducción del tiempo de desarrollo de simulaciones distribuidas basadas en HLA, habiendo sido probado en distintos ambientes como, por ejemplo, el proyecto "experiencia de exploración simulada" de la NASA.

Existen además alternativas que combinan *lenguajes de programación* con soporte específico a nivel de *sistema operativo* para *arquitecturas de computadoras avanzadas*. Este es el caso de Steinman (2005) donde se aborda la propuesta del sistema de Georgia Tech Time Warp y el entorno de desarrollo WarpIV. Investigaciones complementarias se abocaron al diseño de algoritmos de simulación distribuida para arquitecturas de alto rendimiento (Liu, 2013).

Otras investigaciones vinculadas se han llevado a cabo a fin de balancear la carga a procesar (Alghamdi, De Grande & Boukerche, 2015) y gestionar la cantidad de datos transferidos entre simuladores (Raczy, Tan & Yu, 2005). También se han desarrollado tecnologías específicas basadas en agentes, combinando computación distribuida y cómputo de alto rendimiento; como REPAST HPC (Collier, Ozik & Macal, 2015) y PDES-MAS (Suryanarayanan, Theodoropoulos & Lees, 2013). Además, se han propuesto enfoques generales para simulaciones distribuidas híbridas basadas en agentes y eventos discretos (Anagnostou & Taylor, 2017).

Como enfoque complementario, algunas investigaciones han aplicado avances de la Ingeniería de Software para simulaciones distribuidas. El trabajo de Bocciarelli, D'Ambrogio & Fabiani (2012) utiliza el lenguaje System Modeling Language (SysML) como herramienta de soporte a la creación de simulaciones distribuidas. Recientemente, estos autores han propuesto un enfoque dirigido por modelos para producir componentes de simulación que respeten el estándar HLA a partir de modelos SysML (Bocciarelli et al., 2019). Investigaciones similares se han realizado en virtud de definir la estructura de simulación distribuida por medio del uso de patrones de software (Möller, Antelius, Johansson & Karlsson 2016) y procesos de negocio colaborativos (Falcone, Garro, D'Ambrogio & Giglio, 2018).

Independientemente de la forma en la cual se genera la simulación distribuida, las nuevas tecnologías de computación disponibles han dado lugar a avances tecnológicos en el área. En este caso, se destacan: *i)* arquitecturas basadas en servidores (Al-Zoubi & Wainer, 2013), y *ii)* uso de computación en la nube como herramienta de trabajo (Zehe, Knoll, Cai & Aydt, 2015). Estos trabajos han fomentado el desarrollo del "Modelado y Simulación como Servicio" (*Modeling and Simulation as a Service*), a fin de definir servicios de simulación con premisas equivalentes a las usadas en contextos de computación en la nube (Blas, Leone & Gonnet, 2019). En esta área, los desarrollos más relevantes son la plataforma de simulación CloudSME (Taylor et al., 2014) y los estándares para la implementación de estos servicios en la nube (Siegfried, Van Den Berg, Cramp & Huiskamp, 2014).

3.1. Simulación Distribuida para el Desarrollo de Experimentos

La ejecución de múltiples experimentos sobre un modelo de simulación representa un desafío especial para la simulación distribuida ya que no sólo se debe coordinar la ejecución sino también los resultados (recopilación, análisis y visualización). La interacción se presenta en la comunicación desde/hacia el gestor de experimentos.

En los últimos diez años, las aplicaciones que brindan soporte a este tipo de simulación distribuida se han centrado en dos alternativas: *i)* uso de recursos de procesamiento fijos (como, por ejemplo, una red de computadoras), y *ii)* uso de recursos de computación en la nube bajo demanda. En el primer caso, se destacan el sistema WINGRID diseñado para acelerar las simulaciones de riesgo crediticio en bancos europeos (Mustafee & Taylor, 2009) y el uso de redes de computadoras para la simulación de las vías bioquímicas en el cáncer (Liu et al., 2014). Por su parte, en el segundo caso se destacan dos plataformas de desarrollo: JADES (Rak, Cuomo & Villano, 2012) y CloudSME (Taylor et al., 2014). La plataforma JADES fue adaptada para ejecutar en paralelo y en la nube, simulaciones basadas en agentes. La plataforma CloudSME, en cambio, ha sido utilizada para ejecutar experimentos de simulación en múltiples nubes. Una experiencia de éxito reciente ha sido la del uso del servicio en la nube SEMSim (Zehe, Knoll, Cai & Aydt, 2015) que brinda soporte para el desarrollo y experimentación de simulaciones de sistemas de tráfico urbano de gran escala.

4. Aplicaciones Actuales y Tendencias

A fin de analizar los dominios en los cuales se aplica simulación distribuida en la actualidad, se presentan los resultados obtenidos a partir de la revisión de trabajos de investigación publicados. La búsqueda se realizó con la base de datos digital *ScienceDirect* utilizando el término *Distributed Simulation*[2] como indicador de título, resumen, y palabras clave, restringiendo el período de publicación entre los años 2010 y 2019. Como resultado se obtuvieron 258 trabajos los cuales se analizaron manualmente con el objetivo de desestimar resultados falsos positivos.

Del análisis realizado sobre los trabajos restantes (138), se identificaron nueve áreas de aplicación principales, a saber: *defensa*, *medio ambiente*, *energía*, *arquitectura de computadoras y sistemas de control*, *salud*, *manufactura y cadenas de suministro*, *redes de comunicación*, *tecnología espacial* y *transporte/tráfico*. Para cada área, se evidenció un notorio énfasis en relación al tipo de modelo de simulación (basado en agentes -ABS, eventos discretos -DES-, pasos del tiempo -timestepped-, continuo, tiempo real e hibrido), su objetivo (modo A, B o C), el enfoque de administración de tiempo (gestión de tiempo -conservador u optimista- o tiempo real) y la tecnología soporte (tradicional, cloud, grid, HLA, arquitectura orientada a servicios -SOA-, supercomputadores -SC). La Tabla 1 resume los resultados obtenidos por área, los cuales han sido ordenados según el porcentaje de publicaciones analizadas respecto al total de trabajos[3].

[2] Se optó por utilizar el término en inglés dada la naturaleza de los artículos científicos.

[3] En esta sección se indican algunos de los trabajos más relevantes de cada área. Dada la cantidad de trabajos analizados resulta imposible incluir todos como bibliografía.

Como puede observarse, la mayoría de los trabajos de simulación distribuida refieren a *manufactura y cadena de suministro*. Con mayor frecuencia la técnica es aplicada para la vinculación de distintos modelos independientes en nuevos modelos de mayor tamaño (Hibino, Fukuda & Yura, 2015), aunque también se han encontrado casos de entrenamiento distribuido (Silvente, Zamarripa & Espuña, 2012). Aún más, dado que en esta área la aplicación de simulación distribuida se encuentra más difundida, la revisión bibliográfica dio como resultado trabajos vinculados a desarrollos complementarios para su aplicación, como ser: representación de datos (Lin, Wang, Hu & Long, 2012; Long, 2014), ontologías como soporte al reúso de modelos (Sarli, Leone & Gutiérrez, 2016), y adaptaciones de HLA para facilitar su uso en este contexto (Strassburger & Taylor, 2012).

La tendencia en el sector *defensa* (militar) se centra en el uso de simulación distribuida como vehículo para el desarrollo de entrenamientos en tiempo real de forma distribuida (Zhang et al. 2019). Al igual que en el caso anterior, se han propuesto mejoras de HLA y estándares asociados (Wang, Gao, Wei, & Yin, 2012).

Área	Tipo de Modelo	Objetivo	Enfoque	Tecnología	#Publicaciones
Manufactura y cadenas de suministro	DES	B y C	Gestión de tiempo	Tradicional HLA Grid Cloud	40 (28,99%)
Defensa	Tiempo Real Hibrida (Tiempo Real y DES)	B	Tiempo Real	HLA Cloud SOA	20 (14,49%)
Medio ambiente	Hibrida (DES, ABS, Continua)	B	Gestión de tiempo	Tradicional	14 (10,14 %)
Energía	ABS Hibrida (ABS, Continua)	A, B y C	Gestión de tiempo	Tradicional Cloud	13 (9,42%)
Transporte/ Tráfico	Tiempo Real	B	Gestión de tiempo	Tradicional Cloud	12 (8,70%)
Redes de comunicación	DES	A	Gestión de tiempo	Tradicional	11 (7,97%)
Tecnología Espacial	DES Timestepped	B	Tiempo Real	Tradicional	11 (7,97%)
Arquitectura de computadora y sistemas de control	DES Timestepped Hibrida (DES y Continua	A y B	Gestión de tiempo	Tradicional SC	9 (6,52%)
Salud	DES Hibrida (DES/ ABS)	A, B y C	Gestión de tiempo	Tradicional Cloud	8 (5,80%)

Tabla 1 – Análisis de los resultados obtenidos de la revisión bibliográfica agrupados por área.

Por su parte, tanto en el área de *medio ambiente* como en la de *energía* se encontraron múltiples trabajos que aplican simulación distribuida bajo un formato híbrido. En *medio ambiente*, el objetivo suele ser vincular modelos desarrollados con diferentes enfoques en una única simulación con el fin de analizar problemas ambientales (Ma et al., 2019). En cambio, en *energía* los trabajos vinculan modelos (típicamente discretos y continuos) para el análisis de alternativas de diseño en sistemas de energía (Bottura et al., 2013). En este último caso, también se encontraron simulaciones basadas en agentes para analizar redes eléctricas (Perkonigg, Brujic, & Ristic, 2013) junto con experimentación distribuida en base a modelos de redes eléctricas inteligentes basadas en esquemas en la nube (Anderson, Du, Narayan & Gama, 2014).

Las aplicaciones de simulación distribuida en el área de *transporte* y *tráfico* van desde el estudio de diseños para sistemas de transporte en ciudades inteligentes (Xu, Aydt, & Lees, 2012), hasta el análisis y predicción del tráfico (De Grande, Boukerche, Guan, & Aljeri, 2016). En *redes de comunicación*, la simulación distribuida se utiliza comúnmente para acelerar grandes modelos de redes como ser, por ejemplo, redes con sensores inalámbricos y redes ad-hoc móviles (Huang, Alexopoulos, Hunter, & Fujimoto, 2012). Por su parte, en el área de *tecnología espacial* las aplicaciones de simulación distribuida se han centrado en el reúso de modelos y su ejecución en tiempo real (Rabelo et al., 2013), por ejemplo, para establecer bases de exploración lunar (Falcone, Garro, Longo, & Spadafora, 2014).

En relación al área de *arquitecturas de computadoras y sistemas de control*, las aplicaciones de simulación distribuida abordan problemas vinculados a sistemas ciber-físicos y embebidos (Garraghan et. al., 2016). Además, se ha abordado el estudio de diseños alternativos para supercomputadoras (Liu, 2013).

Finalmente, en el área de *salud* se encontraron muy pocos trabajos vinculados al uso de simulación distribuida. En líneas generales, el objetivo suele ser uso de simulación para entrenamiento médico o quirúrgico (Khan et al., 2015).

4.1. Oportunidades, Nuevos Desafíos y Tecnologías de la Información

Aunque la simulación distribuida existe desde hace varios años, la evolución de las tecnologías de la información ha fomentado su aplicación en múltiples dominios. En este contexto, algunas de las oportunidades identificadas como disparador de nuevos desarrollos son:

- *Uso de computación en la nube para acelerar los tiempos de experimentación*: Normalmente la experimentación queda limitada por el tiempo disponible (es decir, se determina una cierta cantidad de tiempo para experimentar sobre los modelos previo a su uso efectivo). En este contexto, el uso de computación en la nube como herramienta alternativa resulta muy atractivo ya que permite adquirir recursos de procesamiento bajo demanda según el tiempo disponible para realizar la experimentación.
- *Estudio de sistemas emergentes a gran escala (personalización en masa, cadenas de suministro complejas, sistemas ciber-físicos de la industria 4.0 y las ciudades inteligentes)*: Se deben adaptar los enfoques de simulación para manejar modelos extensos con gran cantidad de datos junto con la ejecución de análisis refinados por medio de la experimentación.

- *Construcción de ciber-infraestructuras escalables y flexibles para la comunidad de investigación que mejoren el rendimiento de los sistemas para simulación*: Es necesario construir infraestructuras que faciliten la vinculación y el reúso de modelos, facilitando tanto la creación de simulaciones de grandes sistemas como así también su ejecución.

Un punto no menos importante es que, de los trabajos analizados, se observa que las herramientas de simulación utilizadas corresponden (en la mayoría de los casos) a desarrollos propios, aplicaciones de código abierto o librerías. Existen muy pocas aplicaciones de software para la implementación de simulaciones distribuidas que se correspondan con paquetes de simulación comerciales.

4.2. Importancia del Estándar HLA

Uno de los avances con mayor potencial para su uso en aplicaciones de simulación distribuida es HLA. Esta tecnología es un estándar de facto que se utiliza en la industria pero que, dada su complejidad, requiere de una gran cantidad de conocimiento y habilidades técnicas específicas para su aplicación. Este estándar se encuentra diseñado específicamente para resolver problemas de interoperabilidad y reusabilidad entre componentes de software. Sin embargo, ha sido utilizado con éxito para gestionar simulaciones distribuidas ya que define servicios que permiten crear una simulación global compuesta por múltiples participantes distribuidos.

Para componer la simulación global usando HLA, los participantes deben cumplir las especificaciones del estándar. Los participantes HLA se denominan *federados*. Los *federados* se agrupan en una *federación* a efectos de representar la totalidad de la simulación. Los *federados* que conforman una *federación* se comunican por medio de un *RTI* (*Run Time Infrastructure*). Este *RTI* gestiona la *federación*, autoriza la comunicación entre *federados*, y provee servicios de administración de tiempo, objetos y datos, entre otros. Para lograr el funcionamiento de una *federación*, se requiere un *FOM* (*Federation Object Model*) para la formalización de las interacciones y comunicaciones entre los *federados* de la *federación*.

La formulación del estándar garantiza un correcto intercambio de información entre los componentes (*FOM*, *federación*, *RTI* y *federado*), lo que contribuye a gestionar un sistema global distribuido. Luego, mediante el uso de los servicios del estándar, distintas aplicaciones pueden interoperar y ser reutilizadas para componer sistemas más grandes. A partir de estas características, es posible construir sistemas que conecten distintos tipos de componentes en virtud de formar una simulación distribuida: modelos de simulación (con comportamientos específicos), componentes de visualización, componentes de registro de eventos (o auditoría), componentes de gestión de estado del sistema, y componentes de control (o corrección) del sistema.

En este sentido, el estándar HLA sienta las bases necesarias para la gestión distribuida requerida como base de trabajo en este tipo de simulaciones. Por ejemplo, Sarli, Leone & Gutiérrez (2018) presentan un marco conceptual basado en ontologías que brinda soporte a la construcción de simulaciones distribuidas para cadenas de suministro usando HLA como base para su definición.

5. Conclusiones

En términos generales, la aplicación de simulación distribuida como contraparte de la simulación tradicional depende, en mayor medida, del conocimiento del grupo de trabajo. Sin embargo, en los últimos años con el surgimiento de las nuevas tecnologías de la información se ha dado un avance importante en su aplicación como mecanismo de estudio en distintas áreas. Incluso cuando las técnicas de simulación distribuida pueden resultar complejas dado su modo, enfoque de administración de tiempo y tecnología soporte, los casos de éxito analizados a partir de la literatura evidencian una tendencia creciente en su aplicación.

De la revisión bibliográfica realizada, se observa que la mayoría de los trabajos vinculados al área de *manufactura y cadenas de suministro* abordan el estudio de sistemas teóricos de gran complejidad a fin de compatibilizar gran cantidad de modelos e información en una única simulación. Este es el motivo por el cual se observa un evidente incremento de investigaciones científicas en el área. En cambio, en áreas como *defensa* y *transporte*, los trabajos publicados suelen abordar aplicaciones de simulación distribuida en casos reales. Esto conlleva a que, en la mayoría de los casos, las soluciones de casos reales apliquen estrategias de tiempo real. En contraposición, los contextos científicos optan normalmente por trabajar con gestión de tiempo conservador u optimista.

La investigación realizada ha demostrado que, con el paso del tiempo, al mejorar las tecnologías de la información que brindan soporte a las herramientas de desarrollo, se ha dado un incremento en la aplicación de esta técnica. Aún más, con el surgimiento de nuevos problemas ingenieriles abocados a la experimentación de diseños, sistemas y/o situaciones complejas en base a simulación distribuida (ciudades inteligentes, redes de comunicación e industria 4.0; entre otros), se ha dado un incremento de investigaciones complementarias. En este último caso, la representación del conocimiento y el desarrollo/adaptación de estándares actúan como mecanismos claves para compatibilizar tecnologías. Además, por medio de la aplicación de técnicas de Ingeniería de Software, es posible desarrollar herramientas de software que contribuyan a disminuir los conocimientos y habilidades técnicas requeridas para, por ejemplo, la construcción de simulaciones distribuidas basadas en HLA. En este sentido, el uso de las nuevas tecnologías de la información (como ser, por ejemplo, la computación en la nube) como soporte a la simulación distribuida es aún materia de investigación y debe ser atendido por la comunidad a fin de contribuir en el estudio de nuevos dominios.

Referencias

Al-Zoubi, K., & Wainer, G. (2013). RISE: A general simulation interoperability middleware container. *Journal of Parallel and Distributed Computing*, 73(5), 580–594.

Alghamdi, T., De Grande, R., & Boukerche, A. (2015). Enhancing load balancing efficiency based on migration delay for large-scale distributed simulations. In *Proceedings of the IEEE/ACM 19th international symposium on distributed simulation and real time applications*, (pp 33–40). New York, NY: IEEE.

Anderson, K., Du, J., Narayan, A., & Gamal, A. (2014). GridSpice: A distributed simulation platform for the smart grid. *IEEE Transactions on Industrial Informatics*, 10(4), 2354–2363.

Anagnostou, A., & Taylor, S. (2017). A Distributed Simulation Methodological Framework for OR/MS Applications. *Simulation Modelling Practice and Theory*, 70 (1), 101–119.

Blas, M., Leone, H., & Gonnet, S. (2019). Modelado y Verificación de Patrones de Diseño de Arquitectura de Software para Entornos de Computación en la Nube. *RISTI - Revista Ibérica de Sistemas e Tecnologias de Informação*, 35, 1-17.

Bocciarelli, P., D'Ambrogio, A., & Fabiani, G. (2012). A model-driven approach to build HLA-based distributed simulations from SysML models. In *Proceedings of the 2012 international conference on simulation and modeling methodologies, technologies and applications*, (pp 49–60). INSTICC.

Bocciarelli, P., D'Ambrogio, A., Falcone, A., Garro, A., & Giglio, A. (2019). A model-driven approach to enable the simulation of complex systems on distributed architectures. *SIMULATION*, 95(12), 1185–1211.

Bottura, R., Babazadeh, D., Zhu, K., Borghetti, A., Nordström, L., & Nucci, C. (2013). SITL and HLA Co-simulation platforms: Tools for analysis of the integrated ICT and electric power system. In *Proceedings of the 2013 EuroCon conference*, (pp 918–925).

Collier, N., Ozik, J., & Macal, C. (2015). Large-scale agent-based modeling with repast HPC: A case study in parallelizing an agent-based model. *Lecture Notes in Computer Science*, 9523, 454–465. Cham: Springer.

De Grande, R., Boukerche, A., Guan, S., & Aljeri, N. (2016). A modular distributed simulation-based architecture for intelligent transportation systems. *Concurrency and Computation: Practice and Experience*, 28(12), 3409–3426.

Falcone, A., Garro, A., D'Ambrogio, A., & Giglio, A. (2018). Using BPMN and HLA for SoS Engineering: Lessons Learned and Future Directions. In *2018 IEEE International Systems Engineering Symposium*, (pp 1–8). New York: IEEE.

Falcone, A., Garro, A., Longo, F., & Spadafora, F. (2014). Simulation exploration experience: A communication system and a 3D real time visualization for a moon base simulated scenario. In *Proceedings of the 18th IEEE International Symposium on Distributed Simulation and Real-Time Applic*, (pp 113–120).

Falcone, A., Garro, A., Taylor, S., Anagnostou, A., Chaudhry, N., & Salah, O. (2017). Experiences in Simplifying Distributed Simulation: The HLA Development Kit Framework. *Journal of Simulation,* 11(3), 208–227.

Fujimoto, R. (2016). Research Challenges in Parallel and Distributed Simulation. *ACM Transactions on Modeling and Computer Simulation*, 26(4), 1–29.

Garraghan, P., McKee, D., Ouyang, X., Webster, D., & Xu, J. (2016). SEED: A scalable approach for cyber-physical system simulation. *IEEE Transactions on Services Computing*, 9(2), 199–212.

Hibino, H., Fukuda, Y., & Yura, Y. (2015). A synchronization mechanism with shared storage model for distributed manufacturing simulation systems. *International Journal of Automation Technology*, 9(3), 248–260.

Huang, Y., Alexopoulos, C., Hunter, M., & Fujimoto, R. (2012). Ad hoc distributed simulation methodology for open queueing networks. *SIMULATION*, 88(7), 784–800.

IEEE. (2010). *IEEE Standard for Modeling and Simulation (M & S) High Level Architecture (HLA) Framework and Rules*. New York: IEEE.

Khan, R., Aydin, A., Khan, M., Dasgupta, P., & Ahmed, K. (2015). Simulation-based training for prostate surgery. *BJU International*, 116(4), 665–674.

Lin, J., Wang, G., Hu, Z., & Long, Q. (2012). A supply chain architecture based on ontology for distributed simulation and modeling. *International Journal of Digital Content Technology and Its Applications*, 6(5), 225–234.

Liu, E. (2013). A parallel approach of interest management in exascale simulation systems. In *Proceedings of the twelfth international conference on modeling and applied simulation*, (pp 57–60). Rende, Italy.

Liu, X., Taylor, S., Mustafee, N., Wang, J., Gao, Q., & Gilbert, D. (2014). Speeding up systems biology simulations of biochemical pathways using Condor. *Concurrency Computation Practice and Experience*, 26(17), 2727–2742.

Long, Q. (2014). Distributed supply chain network modelling and simulation: Integration of agent-based distributed simulation and improved SCOR model. *International Journal of Production Research*, 52(23), 6899–6917.

Ma, R., Geng, C., Yu, Z., Chen, J., & Luo, X. (2019). Modeling City-Scale Building Energy Dynamics through Inter-Connected Distributed Adjacency Blocks. Energy and Buildings, 202(November), 109391.

Möller, B., Antelius, F., Johansson, M., & Karlsson, M. (2016). Building Scalable Distributed Simulations: Design Patterns for HLA DDM. In *Proceedings of the 2016 Fall Simulation Interoperability Workshop*, (pp 1–10).

Mustafee, N., & Taylor, S. (2009). Speeding up simulation applications using WinGrid. *Concurrency Computation Practice and Experience*, 21(11), 504–523.

Perkonigg, F., Brujic, D., & Ristic, M. (2013). MAC-Sim: A multi-agent and communication network simulation platform for smart grid applications based on established technologies. In *Proceedings of the 2013 IEEE international conference on smart grid communications*, (pp 570–575).

Powers, M., Sanchez, S., & Lucas, T. (2012). The Exponential Expansion of Simulation in Research. In *Proceedings of the 2012 Winter Simulation Conference*, (pp 138:1–138:12). Huntington Beach, CA: IEEE Press.

Rabelo, L., Sala-Diakanda, S., Pastrana, J., Marin, M., Bhide, S., … Joledo, O. (2013). Simulation modeling of space missions using the high level architecture. *Modelling and Simulation in Engineering*, 2013 (11).

Raczy, C., Tan, G., & Yu, J. (2005). A sort-based DDM matching algorithm for HLA. *ACM Transactions on Modeling and Computer Simulation*, 15(1), 14–38.

Rainey, L., & Tolk, A. (2015). *Modeling and Simulation Support for System of Systems Engineering Applications*. Hoboken, NJ: John Wiley & Sons.

Rak, M., Cuomo, A., & Villano, U. (2012). mJADES: Concurrent simulation in the cloud. In *Proceedings of the 2012 international conference on complex, intelligent, and software intensive systems*, (pp 853–860). New York: IEEE.

Sarli, J., Leone, H., & Gutiérrez, M. (2016). Ontology-based semantic model of supply chains for modeling and simulation in distributed environment. In *Proceedings of the 2016 winter simulation conference*, (pp 1182–1193).

Sarli, J., Leone, H., & Gutiérrez, M. (2018). SCFHLA: Un Modelo de Interoperabilidad Semántica para Simulación Distribuida de Cadenas de Suministro. *RISTI - Revista Ibérica de Sistemas e Tecnologias de Informação*, 30, 34-50.

Siegfried, R., Van Den Berg, T., Cramp, A., & Huiskamp, W. (2014). M&S as a service: Expectations and challenges. In *Proceedings of the 2014 fall simulation interoperability workshop*, (pp 248–257). Orlando, FL: SISO.

Silvente, J., Zamarripa, M., & Espuña, A. (2012). Use of a distributed simulation environment for training in supply chain decision making. *Computer Aided Chemical Engineering*, 30(2012), 1402–1406.

Steinman, J. (2005). The WarpIV simulation kernel. In *Proceedings of the 2005 IEEE Workshop on Principles of Advanced and Distributed Simulation*, (pp 161–170). New York, NY: IEEE.

Strassburger, S., & Taylor, S. (2012). A comparison of the CSPI and CMSD standards. In *Proceedings of the 2012 spring simulation interoperability workshop*, (pp 82–89).

Suryanarayanan, V., Theodoropoulos, G., & Lees M. (2013). PDES-MAS: Distributed Simulation of Multi-agent Systems. *Procedia Computer Science*,18(1), 671–681.

Taylor, S. (2019). Distributed Simulation: State of Art and Potential for Operational Research. *European Journal of Operational Research*, 273(1), 1–19.

Taylor, S, Kiss, T., Terstyanszky, G., Kacsuk, P., & Fantini, N. (2014). Cloud computing for simulation in manufacturing and engineering: Introducing the CloudSME simulation platform. *Simulation Series*, 46(2), 89–96.

Venu, M., & Joe, I. (2014). Improving Performance of Optimistic Simulation for Distributed Simulation System Using Speculative Computation. In *2014 International Conference on Information and Communication Technology Convergence*, (pp 428–32). New York, NY: IEEE.

Wang, Z., Gao, Q., Wei, X., & Yin, S. (2012). Design and implementation of a simulation system for armored communication countermeasure training. *Applied Mechanics and Materials*, 128–129, 1363-1366.

Xu, Y., Aydt, H., & Lees, M. (2012). SEMSim: A distributed architecture for multi- scale traffic simulation. In *Proceedings of the 2012 ACM SIGSIM workshop on principles of advanced and distributed simulation*, (pp 178–180).

Yücesan, E., Luo, Y., Chen, C., & Lee, I. (2001). Distributed Web-Based Simulation Experiments for Optimization. *Simulation Modelling Practice and Theory*, 9(1), 73–90.

Zacharewicz, G., Frydman, C., & Giambiasi, N. (2006). Lookahead Computation in G-DEVS/HLA Environment. *SNE Simulation News Europe*, 16(2), 15–24.

Zehe, D., Knoll, A., Cai, W., & Aydt, H. (2015). SEMSim Cloud Service: Large-scale urban systems simulation in the cloud. *Simulation Modelling Practice and Theory*, 58(2), 157–171.

Zhang, C., Chen, J., Li, Ya., Li, Yu., & Chai, W. (2019). Satellite group autonomous operation mechanism and planning algorithm for marine target surveillance. *Chinese Journal of Aeronautics*, 32(4), 991-998.