

Hacia una Metodología para la Construcción de Modelos de Simulación Basados en Sistemas Multi-Agente

Julián Moreno Cadavid, Juan David Velásquez Henao
y Demetrio Ovalle Carranza

UNIVERSIDAD NACIONAL DE COLOMBIA. Escuela de Sistemas.
Grupo de Investigación y Desarrollo en Inteligencia Artificial - *GIDIA*
{jmoreno1; jdvelasq; dovalle}@unalmed.edu.co

Recibido para revisión Octubre 2005, aceptado Octubre 2005, versión final recibida Noviembre 2005

Resumen: El objetivo de este artículo es presentar las diferentes fases, modelos y artefactos de una metodología creada por el grupo *GIDIA* (Grupo de Investigación y Desarrollo en Inteligencia Artificial) para el desarrollo de modelos de simulación basados en Sistemas Multi-Agente. La metodología propuesta se fundamenta en algunos de los aspectos más sobresalientes de la ingeniería de software clásica, de la ingeniería de software orientada a agentes y la simulación de sistemas. Al integrar dichos aspectos se obtiene una metodología robusta y bien formalizada, la cual podrá ser aplicada en modelos de simulación sociales, económicos, políticos, ambientales, etc.

Palabras Clave: Simulación basada en Sistemas Multi-Agente, Metodologías de Desarrollo de Modelos.

Abstract: The aim of this paper is to show the different phases, models and artifacts of a methodology that was created by the research group in order to develop multi-agent based simulation models. The proposed methodology is based on some of the most outstanding features of the classic software engineering, the agent-oriented software engineering and the systems simulation. Once those features have been integrated we obtain a robust and well formalized methodology which could be applied in simulating social, economic, political, or even environmental models.

Keywords: Multi-Agent based Simulation, Model Development Methodologies.

1 INTRODUCCIÓN

A pesar de que los modelos de simulación basados en agentes (MABS por sus siglas en inglés) han sido usados con éxito durante la última década para entender diversos procesos sociales así como para asistir la formulación y evaluación de políticas en diferentes sistemas [Conte, Gilbert y Sichman (1998)], ha habido pocos esfuerzos para formalizar, sistematizar y comunicar métodos para su desarrollo [Ramanath y Gilbert (2003)]. Esto se debe principalmente a que MABS es aún un área de estudio reciente, donde existe una marcada falta de acuerdos sobre los acercamientos, técnicas y herramientas para su desarrollo.

Por tal motivo, en este artículo se expondrán diversos enfoques y sus correspondientes metodologías para la creación de proyectos de este tipo. Se describirán las fortalezas y debilidades de cada una de éstas para pos-

teriormente, en base a ellas, proponer una metodología que recoja sus puntos más sobresalientes, de manera que el resultado sea una metodología robusta y bien formalizada.

2 ENFOQUES Y METODOLOGÍAS EXISTENTES

Algunas de las metodologías propuestas para el desarrollo de MABS tienen sus bases en campos diversos como la simulación de sistemas, la ingeniería de software y la ingeniería de software orientada a agentes; cada uno de los cuales con su propio enfoque sobre las fases que deben contener y sobre los artefactos utilizados para llevarlas a cabo.

La simulación de sistemas, y más específicamente la simulación basada en modelos computacionales, consiste en una intersección entre las ciencias sociales, matemáticas y de la computación, que busca la creación de mo-

delos simplificados de la realidad estudiada. Para el desarrollo de tales modelos se siguen los pasos genéricos de conceptualización, diseño, implementación, verificación, validación y publicación de resultados. El inconveniente de este esquema es que es demasiado general y no presenta una formalización sobre los artefactos (entiéndanse los modelos, diagramas, plantillas, etc.) que deben utilizarse en cada fase.

La ingeniería de software por su parte, se fundamenta en una aproximación denominada ciclo de vida del software, el cual es un paradigma universalmente aceptado y que se compone de las fases genéricas de: conceptualización o especificación, análisis, diseño, implementación o construcción, transición, producción y mantenimiento [Jiménez (2003)].

La realización de dichas fases puede ser vista como un proceso retroalimentado, donde la elaboración de una fase puede producir cambios en otra anterior. Este paradigma sin embargo, ha sido criticado por ser demasiado "duro" para la creación de sistemas caracterizados por emular actividades humanas [Checkland y Scholes (1990)] pues se centra en la sistematización de procesos, dejando de lado muchas de las ambigüedades inherentes a los procesos sociales.

Por último, la ingeniería de software orientada agentes (AOSE, por sus siglas en inglés) es un paradigma relativamente reciente que usa la noción de agente como entidad de abstracción primaria [Jennings y Wooldridge (2000)], y en el cual nuevas metodologías y plataformas han ido surgiendo a medida que recibe más atención por parte de la comunidad científica.

Entre dichas metodologías algunas de las más utilizadas son: MAS-CommonKADS [Iglesias (1998)], GAIA [Wooldridge, Jennings y Kinny (2000)], Styx [Bush, Cranefield y Purvis (2001)] y MaSE [DeLoach (2001)], aunque hay también algunos trabajos que proponen la combinación de varias de estas como es el caso de Cernuzzi y Zambonelli (2005). Para el desarrollo de proyectos de sistemas multi-agente (MAS, por sus siglas en inglés) dichas metodologías comparten las fases genéricas de análisis y de diseño, siendo algunas más extensas al considerar una fase previa de conceptualización y, en el caso de aquellas que están asociadas a una plataforma, una fase posterior de implementación.

Una ventaja que presentan estas metodologías para el desarrollo de MABS, es que permiten una representación en términos de agentes de las entidades modeladas, incluyendo aspectos claves de estos sistemas como la organización, el razonamiento, los mecanismos de comunicación y de coordinación, entre otros. Sin embargo, tienen como inconveniente desde el punto de vista de MABS, que son más orientados al desarrollo de sistemas de software distribuidos que para modelos de simulación.

3 METODOLOGÍA PROPUESTA

A manera de resumen, las fortalezas de los tres enfoques analizados en la sección anterior son: el enfoque sistémico y la consideración de las fases de verificación y validación de los modelos de simulación; la formalización, el uso de estándares y el proceso de desarrollo en espiral de las metodologías de creación de software; y la orientación a agentes (junto con las repercusiones que esto representa) de AOSE.

Sin embargo, dentro de dichos enfoques, no existe una metodología única que cuente con todos los requerimientos necesarios para la construcción de proyectos MABS. Por tal motivo, en este artículo se propone una metodología robusta que recoge las ventajas de cada uno de estos enfoques así como algunos de los artefactos utilizados en las diferentes fases que algunas de sus metodologías proponen.

El esquema de las fases de la metodología propuesta así como de los artefactos empleados en cada una, se presenta en la Figura 1.

3.1 Fase de Captura de Requerimientos

Esta fase se considera una fase previa al proceso de modelamiento en la que se recopila la información relacionada con el dominio del sistema a modelar. Dicha información debe ser lo más completa posible, puesto que es el punto de partida para la realización de las fases posteriores y puede provenir de una o varias fuentes como: entrevistas con expertos del dominio, cuestionarios, informes técnicos y/o conceptuales, diagramas de procesos, entre otros.

3.2 Fase de Conceptualización

Esta fase busca brindar un contacto inicial con el problema de estudio. Consiste en explorar y delimitar dicho problema y elaborar un primer esbozo del sistema que puede resolverlo. En esta fase se identifican las entidades que hacen parte del sistema real, así como sus objetivos, tareas e interacciones; y de manera global, el funcionamiento del sistema.

En la metodología propuesta se hace uso del análisis centrado en el usuario, propio de las metodologías orientadas a objetos [Jacobson, Christerson, Jonsson y Övergaard (1992)], y cuyo principal objetivo es comprender las necesidades de los usuarios respecto al sistema en construcción. Para el caso de MABS, el usuario al que se hace referencia sería el experto en el dominio sobre el que se hace el modelo (o el modelador, en caso que éste cuente con dicho conocimiento); mientras que las necesidades o requisitos de los cuales se habla pueden traducirse en la recopilación de las características y funcionalidad que debe poseer el sistema en desarrollo para

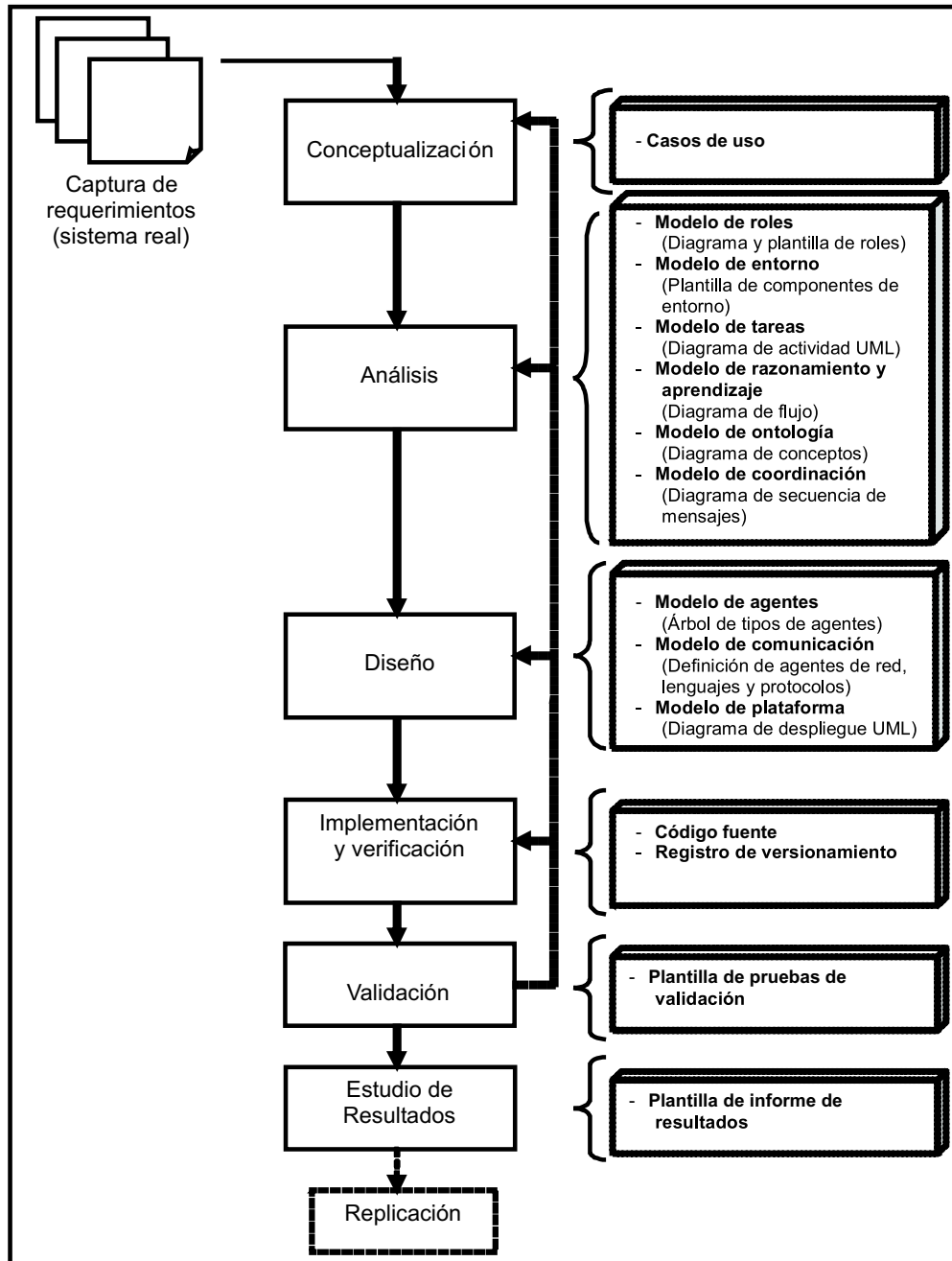


Figura 1: Esquema de desarrollo de la metodología propuesta

simular su contraparte real.

- Casos de Uso** Para esta fase se propone la utilización de casos de uso [Rumbaugh (1995)] común para las metodologías de la ingeniería de software y utilizados en la mayoría de metodologías orientadas a agentes. Este modelo describe las relaciones que se presentan entre las entidades que hacen parte del sistema analizado a un nivel de abstracción elevado, es decir, sin muchos detalles. En este modelo dichas entidades se denominan actores y las interacciones se denominan usos. El artefacto utilizado se deriva de la notación gráfica propuesta por Jacobson et al. (1992), presentada en la Figura 2, y se complementa con una plantilla como se muestra en la Tabla 1.

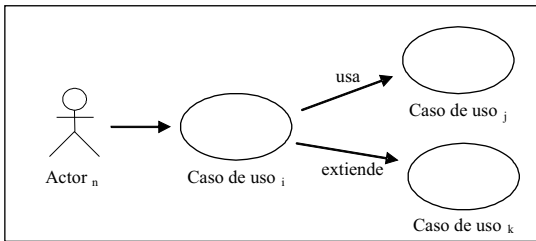


Figura 2: Notación gráfica de los casos de uso

Tabla 1: Plantilla de caso de uso

Actores involucrados	
Objetivos Asociados	
Descripción	
Precondiciones	
Flujos alternativos	
Poscondiciones	

3.3 Fase de Análisis

El objetivo de esta fase es la construcción de modelos conceptuales que permiten representar tanto la estructura como el comportamiento del sistema. Un modelo conceptual se utiliza para especificar el "qué" del espacio del problema, sin importar todavía el "cómo" se llegará a la solución, lo que lo hace independiente de la implementación. Esta fase es común para la ingeniería de software y para AOSE, mientras que no es considerada en el esquema de la simulación de sistemas en la que el "qué" y el "cómo" se modelan de manera conjunta en la fase de diseño.

Para el desarrollo de esta fase se hace uso en la metodología propuesta de varios modelos que buscan representar diferentes aspectos del sistema, de manera que todos se complementan para dar una visión general de éste.

- Modelo de Roles** El modelo de partida en esta fase es el modelo de roles, en el que se identifican,

en base a los actores previamente definidos en la fase anterior, cuáles van a ser los roles existentes en el sistema. El concepto de rol puede entenderse como una descripción abstracta del papel llevado a cabo por cada entidad, el cual tiene asociado una serie de responsabilidades (o funciones) dentro del sistema, así como un conjunto de capacidades para cumplirlas.

Para este modelo se utiliza como artefacto un diagrama de roles propio de la metodología propuesta, como el que se muestra en la Figura 3. En éste se representan los roles identificados por medio recuadros, así como las interacciones entre ellos: una línea sencilla denota una comunicación/colaboración, una flecha denota una relación de autoridad/jerarquía, y una línea con un triángulo al final denota una relación de herencia (donde el rol que se encuentra en el extremo del triángulo puede entenderse como una subclase/especialización del otro). En este modelo también se utiliza una plantilla de roles como la que se muestra en la Tabla 2 para complementar el diagrama anterior, donde se presenta por cada rol la descripción de su papel dentro del sistema, sus objetivos y las responsabilidades que tiene asignadas así como las capacidades con las que cuenta y la información que necesita para realizarlas.

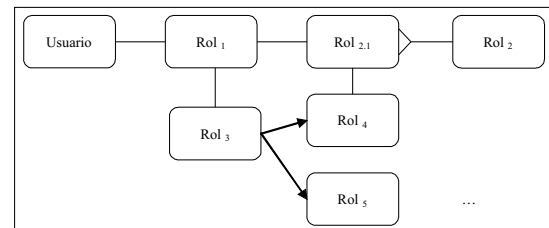


Figura 3: Diagrama de roles

Tabla 2: Plantilla de rol

Rol: <nombre del rol>
Objetivos: <descripción textual>
Responsabilidades: <Responsabilidad 1> <Responsabilidad 2> ... <Responsabilidad n>
Capacidades: <Capacidad 1> <Capacidad 2> ... <Capacidad n>
Información requerida: <Información 1> <Información 2> ... <Información 1>

- Modelo del Entorno** En este modelo se especifica el entorno dentro del cual interactúan los roles del sistema, es decir, se describen los componentes del sistema que no son agentes, pero que de alguna forma intervienen con los procesos llevados a cabo por estos.

Tal intervención puede entenderse como un flujo de información uni o bidireccional entre los roles y el entorno en el que se puede identificar las entradas, en caso que estas sean necesarias; y las salidas, si es que hay alguna. El proceso mediante el cual tal información es procesada también se describe en este modelo por medio de la plantilla textual, a menos que este sea una "caja negra", o sea irrelevante en el modelo de simulación. El artefacto utilizado en este modelo es la plantilla de componentes de entorno, como la que se presenta en la Tabla 3.

Tabla 3: Plantilla de componentes de entorno

Componente: <nombre>	
Entradas	
Dato	Emisor(es)
<ul style="list-style-type: none"> <descripción> <descripción> 	<nombre del rol o componente>
Procesos	
<descripción del procesamiento de la información>	
Salidas	
Dato	Receptor(es)
<ul style="list-style-type: none"> <descripción> <descripción> 	<nombre del rol o componente>

- Modelo de Tareas** Este modelo sirve para detallar la funcionalidad de cada rol y consiste en descomponer y describir cada una de las tareas llevadas a cabo por estos como un secuenciamiento de actividades, tomando en consideración sus capacidades, la información que necesitan y su interacción con el entorno. Para este modelo se utilizan los diagramas de actividad [Rumbaugh, Blaha, Premerlani y y Hedí (1991)] de la notación UML como medio de representación, como se ilustra en la Figura 4.

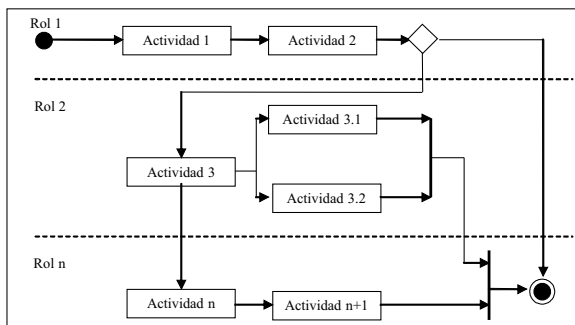


Figura 4: Diagrama de actividad

- Modelo de Razonamiento y Aprendizaje** La función de este modelo es identificar las estructuras de inferencia utilizadas por los roles para llevar a cabo algunas de las actividades que tienen asociadas, bien sea que estas sean de carácter reactivo (respuestas ante estímulos o peticiones provenientes de otros agentes) o proactivo (que sean realizadas de manera autónoma según el juicio del propio agente). Este modelo encapsula las creencias, objetivos, capacidades y decisiones de cada rol, relacionándolas con las actividades que deben llevar a cabo. En este contexto una creencia puede ser entendida como una regla de razonamiento que puede ser alterada según los acontecimientos que se presenten.

El artefacto utilizado para la representación de este modelo es el diagrama de flujo [Tansley y Hayball (1993)] presentado en la Figura 5 donde se esquematiza la lógica de razonamiento para realizar una determinada actividad.

Este modelo es uno de los aportes principales de la metodología propuesta, puesto que no existe un modelo paralelo en las metodologías analizadas que trate de manera formal este aspecto de los agentes.

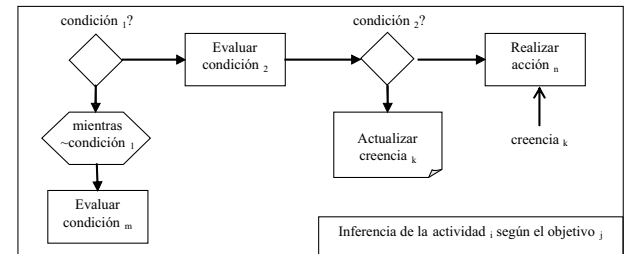


Figura 5: Diagrama de flujo

- Modelo de Ontologías** El objetivo de este modelo es describir la ontología u ontologías del sistema, es decir, la estructura y significado de los principales conceptos agrupados por dominio de aplicación que serán comunicados entre los agentes (independientemente de la forma que en que se empaqueten) y sus relaciones.

La representación de este modelo es similar a la utilizada por el diagrama de clases de UML [Rumbaugh et al. (1991)], donde se presentan los conceptos de la ontología junto con los términos que los componen, tal como se muestra en la Figura 6. Las restricciones de los términos o atributos de cada concepto, junto con una descripción más detallada deben especificarse en una plantilla por separado donde se especifiquen características de los datos como tipo, restricción, obligatoriedad y valor por defecto.

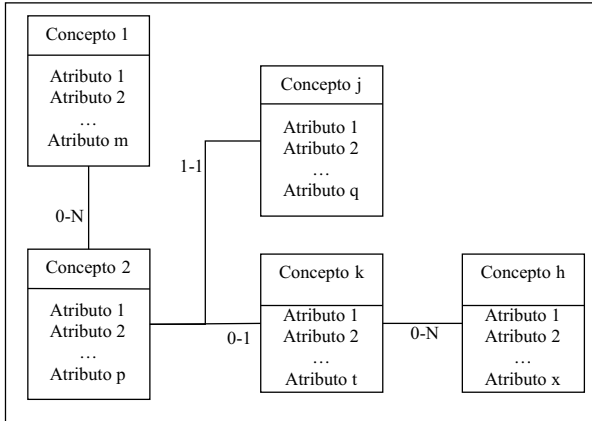


Figura 6: Diagrama de conceptos

- Modelo de Coordinación** Este modelo permite profundizar en las interacciones entre los roles identificando su objetivo, los participantes y las etapas de las que se componen. Dichas interacciones se agrupan en conversaciones o protocolos, que pueden ser vistos como patrones formalizados de comunicación. El artefacto utilizado en este modelo es el diagrama de secuencias de mensajes (MSC, por sus siglas en inglés) [Iglesias (1998)], donde se representa el flujo de intercambio de mensajes agrupado por conversaciones, tal como se muestra en la Figura 7.

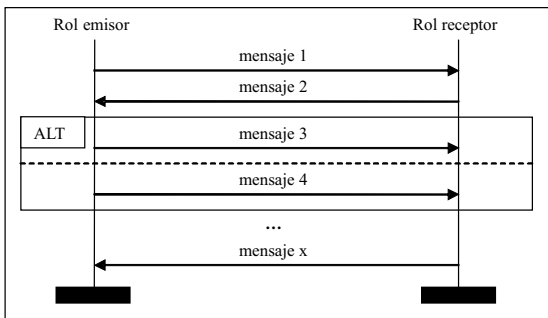


Figura 7: Diagrama de secuencia de mensajes

3.4 Fase de Diseño

Mientras que la fase anterior se encarga del "qué", la fase de diseño se encarga del "cómo" de lo que debe hacer el sistema en desarrollo. Desde el punto de vista de la ingeniería del software, el objetivo de esta fase es transformar los modelos provenientes de la fase de análisis en modelos con un nivel de abstracción lo suficientemente bajo para poder traducirlos posteriormente a código fuente durante la implementación.

Sin embargo, este no es el objetivo principal en el caso de AOSE donde no hay un mapeo uno a uno entre los modelos de ambas etapas. En cambio, lo que se busca

es ampliar la especificación de los requisitos funcionales del sistema e introducir los no funcionales que deben tenerse en cuenta para su implementación, buscando además la independencia del modelo de la plataforma de construcción.

Al igual que en el caso de la fase de análisis, para el desarrollo de esta fase se hace uso en la metodología propuesta de varios modelos que representan diferentes vistas del sistema, capturando ciertos aspectos (ahora más técnicos) que no habían sido considerados.

- Modelo de Agentes** En este modelo se especifican los diferentes tipos de agentes que existirán en el sistema, así como las instancias de dichos agentes durante la ejecución. Se entiende entonces por agente aquella entidad que desempeña uno o mas roles dentro del sistema y por tanto posee las características que éstos especifican (responsabilidades, capacidades, etc.) definidas en la fase anterior.

Para la representación de este modelo se utiliza un árbol de tipos de agentes, en el cual los nodos raíz corresponden a los roles y los demás nodos a los tipos de agentes, presentándose un mapeo lógico entre ambos, como se muestra en la Figura 8. La cardinalidad de las asociaciones se representa con notación $n \cdot \cdot m$, aunque usualmente son relaciones uno a uno.

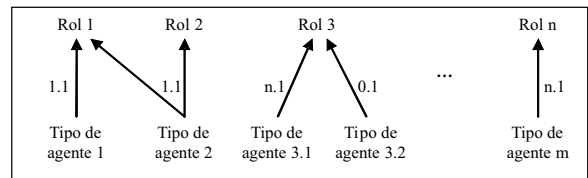


Figura 8: Árbol de tipos de agentes

- Modelo de Comunicación** En este modelo se especifican los requerimientos de las comunicaciones entre los agentes, ya no desde el punto de vista sintáctico y semántico analizados en el modelo de ontología y coordinación respectivamente, si no desde el punto de vista del transporte.

Por una parte, es necesario precisar si se requieren agentes adicionales para administrar la comunicación entre los agentes del sistema (Agentes Servidores de Nombres - ANS, Agentes Facilitadores - DF, Agentes de Canal de Comunicaciones - ACC, Agentes Intermediadores o *Brokers*, Agentes Gestores de Grupos, etc.) algunos de los cuales hacen parte de estándares como FIPA [García (2000)]. Por otra parte, se debe especificar el lenguaje de transporte utilizado, es decir, el lenguaje en el cual se encapsulan los mensajes transmitidos entre los agentes. En este punto se debe definir el estándar (FIPA ACL, KQML, u otro) y la versión utilizada

o, en caso de utilizar uno propio, definir la estructura de la transmisión de mensajes junto con las preformativas utilizadas. Igualmente, es necesario identificar los protocolos de comunicación utilizados durante las conversaciones, en caso que estas se descompongan en uno o varios de estos. En este punto debe definirse si se utilizan protocolos estándar como los de FIPA (ej: *FIPA-ContractNet*, *FIPA-Request*, *FIPA-DutchAuction*, etc.) o si se utilizan unos propios, definir el diagrama de secuencia de estos.

- Modelo de Plataforma** Este modelo permite documentar las decisiones de bajo nivel sobre las características de la plataforma sobre la que se montará el sistema como el lenguaje de implementación seleccionado, el software y hardware empleado, etc. Dicha plataforma de desarrollo puede ser una ya existente como ZEUS, JADE, FIPA-OS, etc. o una plataforma propia.

El artefacto utilizado en este modelo es el diagrama de despliegue de UML, como el que se muestra en la Figura 9, en el que se especifican los módulos que contiene el sistema (agrupamiento de agentes según algún parámetro), su ubicación, su medio de comunicación y la conexión con otros sistemas, repositorios de información o plataformas.

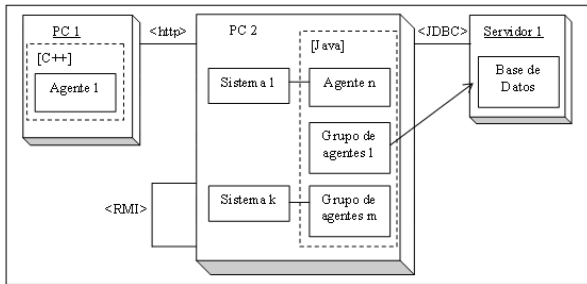


Figura 9: Diagrama de despliegue

3.5 Fase de Implementación y Verificación

Esta fase se compone de dos etapas: En la primera se lleva a cabo la construcción del sistema como tal, es decir, se traducen los modelos de las fases anteriores a código fuente o a una plataforma existente. En esta etapa es importante documentar el proceso de codificación y especificar los detalles técnicos más significativos como el lenguaje o plataforma utilizada, versión, etc. La segunda etapa, conocida como verificación interna, consiste en verificar la correcta correspondencia durante la transformación entre las representaciones abstractas de los modelos de las fases anteriores y el sistema implementado (el modelo de simulación como tal). Es decir, en asegurar que el código fuente generado verdaderamente refleje

el comportamiento implícito de las especificaciones desarrolladas desde el modelo conceptual. En esta etapa es necesario depurar el sistema, preferiblemente usando algunos casos de estudio con resultados predecibles.

En caso tal que el resultado de la verificación obligue a realizar cambios en algún componente del modelo, es necesario llevar un registro de versionamiento en el que se indiquen dichos cambios.

3.6 Fase de Validación

Mientras que en la etapa de verificación de la fase anterior se chequea que el sistema "funcione", en la fase de validación se chequea que "funcione como debe hacerlo". En otras palabras, mientras que la verificación se preocupa porque el sistema desarrollado se ejecute como el modelador espera que lo haga, la validación se preocupa porque el sistema sea un buen reflejo de su contraparte real.

En el caso de MABS particularmente, es necesario que en esta fase la validación se lleve a cabo tanto a nivel micro (a nivel de cada uno de los agentes) como macro (a nivel de la estructura global, entendida como el resultado de las interacciones entre las partes componentes). Esta fase consiste entonces en la realización de una serie de pruebas generales para validar el modelo de simulación, aunque haciendo la salvedad que la aplicación de las mismas dependerá de la naturaleza del modelo. Estas pruebas han sido recopiladas del trabajo de Sterman (2000) y, aún cuando este y otros autores se basan en el enfoque de la dinámica de sistemas para la construcción de los modelos de simulación, las pruebas mencionadas se consideran genéricas para modelos de cualquier tipo.

El artefacto utilizado en esta fase es entonces una tabla resumen con los resultados de cada una de estas pruebas, como la que se presenta en la Tabla 4, donde la columna "Resultado" puede tener uno de los valores: *Satisfactoria*, *Insatisfactoria*, *No aplicable*, y *No concluyente*. Mientras que la columna "Observaciones" contiene información adicional sobre el método utilizado y los resultados obtenidos.

Tabla 4: Plantilla de las pruebas de validación

Prueba	Resultado	Observaciones
Suficiencia de límites		
Consistencia de estructura		
Consistencia dimensional		
Valoración de parámetros		
Condiciones extremas		
Reproducción de comportamientos		
Análisis de sensibilidad		

3.7 Fase de Estudio de Resultados

Como fase final de la metodología, siendo la más importante para el usuario del modelo (sea este o no el mismo modelador), se encuentra la fase de estudio de resultados. En esta fase generalmente se llevan a cabo diferentes análisis de escenarios o casos de estudio, en los que se ejecuta el modelo de simulación dentro de unas condiciones controladas y bajo ciertos parámetros para luego estudiar los resultados obtenidos.

Igual que en la fase de validación, el análisis que se realiza en esta fase se hace tanto a nivel micro como macro, es decir, desde el comportamiento individual de cada agente, hasta el comportamiento emergente del sistema resultado de las interacciones.

El proceso llevado a cabo en esta fase debe seguir pautas del diseño de experimentos y debe especificar de manera clara y concisa las características de los escenarios propuestos junto con los resultados obtenidos y sus correspondientes interpretaciones. En esta fase sin embargo, la metodología propuesta no presenta ningún artefacto para representar el proceso, pero si exhorta, como ya se mencionó, a la presentación de los resultados obtenidos de una manera adecuada, tal como se muestra en la Tabla 5.

Tabla 5: Plantilla de informe de resultados

Escenario	<Descripción>
Características	<Valor de los parámetros> valores fijos distribuciones de probabilidad ... <Supuestos>
Resultados	<Resultados puntuales> tiempo de ejecución periodos de simulación ... <Tablas> variables acumuladoras ... <Graficas> variables vs. tiempo ...
Interpretación	
Conclusiones	

3.8 Fase de Replicación

La mayoría de las fases expuestas anteriormente son usadas para el desarrollo de modelos de simulación de diferente índole. Hay sin embargo, una fase adicional que raramente se encuentra pero que debería ser considerada y es la de replicación (la línea punteada de la Figura 4 denota que esta fase es complementaria).

La replicación se refiere a confirmar que los resultados obtenidos con el modelo son confiables, en el sentido que pueden ser reproducidos por terceros a partir de las abstracciones brindadas por los modelos desarrollados.

4 CONCLUSIONES

La metodología presentada en este artículo es una propuesta, aún en proceso de refinación, para el desarrollo modelos de simulación basados en sistemas multi-agente, y con la cual se intenta llenar de cierta manera parte del vacío que existe en este reciente campo de estudio en cuanto a la falta de metodologías formales. La metodología propuesta se fundamenta en algunos de los aspectos más sobresalientes de la ingeniería de software clásica, de la ingeniería de software orientada a agentes y la simulación de sistemas.

Al integrar dichos aspectos se obtiene una metodología robusta y bien formalizada, la cual podrá ser aplicada en el futuro en modelos de simulación sociales, económicos, políticos, ambientales, etc.

El esquema de esta metodología contempla las fases de captura de requerimientos, análisis, diseño, implementación-verificación, validación, y estudio de resultados; y propone para cada una de estas una serie de modelos y artefactos que brindan diferentes perspectivas del modelo de simulación a desarrollar.

En la actualidad, esta metodología está siendo usada para el desarrollo de modelos de simulación al interior del grupo GIDIA (Grupo de Investigación y Desarrollo en Inteligencia Artificial), pero se espera que luego de refinarla se convierta en una metodología robusta que pueda ser utilizada en otros proyectos tanto académicos como comerciales.

AGRADECIMIENTOS

El trabajo presentado en este artículo fue financiado parcialmente por el proyecto de investigación titulado: "Diseño e implementación de un sistema multi-agentes para la simulación del proceso de negociación en el mercado energético colombiano por medio de contratos normalizados bilaterales" auspiciado por la DIME - Dirección de Investigaciones de la Universidad Nacional de Colombia - Sede Medellín.

REFERENCIAS

- Bush, G., Cranefield, S. y Purvis, M. (2001), 'The styx agent methodology', *The Information Science Discussion Paper Series*.
- Cernuzzi, L. y Zambonelli, F. (2005), Developing mas solutions with gaia and auml, in '31 Conferencia Latinoamericana de Informatica - CLI 2005'.

- Checkland, P. y Scholes, J. (1990), *Soft Systems Methodology in Action*, John Wiley & Sons, Chichester.
- Conte, R., Gilbert, N. y Sichman, J. (1998), Mas and social simulation: A suitable commitment, in 'Multi-Agent-Based Simulation, Second International Workshop, MABS 1998'.
- DeLoach, S. (2001), Analysis and design using MASE and agenttool, in '12th Midwest Artificial Intelligence and Cognitive Science Conference- MAICS 2001'.
- García, D. (2000), Introducción al estándar FIPA, Technical report, Informe técnico UCM-DSIP 98-00. Universidad Complutense de Madrid.
- Iglesias, C. (1998), Definición de una metodología para el desarrollo de Sistemas Multi-Agente, PhD thesis, Universidad Politécnica de Madrid.
- Jacobson, I., Christerson, M., Jonsson, P. y Övergaard, R. (1992), *Object-Oriented Software Engineering. A Use Case Driven Approach.*, ACM Press.
- Jennings, N. y Wooldridge, M. (2000), 'Agent-oriented software engineering', *Handbook of Agent Technology. AAAI/MIT Press* .
- Jiménez, C. (2003), Guía metodológica de desarrollo o adaptación de software, Technical report, Universidad Nacional de Colombia - Sede Medellín.
- Ramanath, A. y Gilbert, N. (2003), 'Towards a methodology for agent-based social simulation research'.
- Rumbaugh, J. (1995), 'Omt: The development model', *JOOP - Journal of Object Oriented Programming* **76**, 8–16.
- Rumbaugh, J., Blaha, M., Premerlani, W. y y Hedí, V. (1991), *Object-Oriented Modeling and Design*, Prentice-Hall.
- Sterman, J. (2000), *Business Dynamics Systems Thinking and Modeling for a Complex World*, McGraw-Hill.
- Tansley, D. y Hayball, C. (1993), *Knowledge-Based Systems Analysis and Design*, BCS Practitioner Series. Prentice-Hall.
- Wooldridge, M., Jennings, N. y Kinny, D. (2000), 'The gaia methodology for agent-oriented analysis and design', *Autonomous Agents and Multi-Agent Systems* **3**, 285–312.