Nota de Investigación N°6

Cormas, una plataforma multiagente para la modelización interactiva

Pierre Bommel[a,b*], Nicolas Becu[c], Christophe Le Page[b,d], François Bousquet[b] y Grégoire Leclerc[b,e]

Octubre 2017

Ideas principales

- La plataforma de modelización multiagente CORMAS (cormas.cirad.fr) es un framework[1] para modelar con facilidad sistemas socioecológicos complejos y simular sus dinámicas.
- Cormas facilita el diseño colectivo de modelos, basándose en los principios establecidos por la comunidad de práctica ComMod (www.commod.org).
- Cormas ofrece nuevas características que permiten la modelización interactiva (o híbrida), es decir, permiten a los actores interactuar con agentes en el proceso de simulación.

Resumen: Esta nota de investigación presenta las nuevas características de Cormas, una plataforma de modelización multiagente dedicada a la gestión de recursos renovables. Cormas es un software libre que tiene como objetivo diseñar fácilmente un Sistema Multi-Agente (SMA) y analizar escenarios. Recientemente Cormas ha adoptado una dirección innovadora orientada al diseño colectivo de modelos y simulación interactiva. Estas simulaciones híbridas mezclan las decisiones tomadas por las partes interesadas y otras por el modelo. Esto le permite interactuar con una simulación cambiando el comportamiento de los agentes y cómo ellos utilizan los recursos. Por lo tanto, es posible explorar colectivamente escenarios de mediano y largo plazo para comprender mejor cómo lograr una situación deseada. A su vez, esto permite una revisión colectiva del modelo conceptual.

Introducción

Cormas es una plataforma de modelización multiagente dedicada a la gestión de recursos renovables [1]. Como *software* libre es utilizado por una comunidad internacional de investigadores [2] que estudia las relaciones entre las sociedades y el ambiente. Cormas está diseñado para facilitar el diseño de Sistemas Multi-Agente (SMA), así como para el monitoreo y análisis de simulaciones. De hecho, la meta de un SMA es entender cómo las entidades independientes pueden interactuar, coordinar y coevolucionar mientras producen efectos en el sistema. Un agente puede ser descrito como una entidad autónoma capaz de adaptarse a un entorno cambiante, asimismo sus acciones pueden cambiar este entorno. Al estar centrado en el individuo, un SMA permite al usuario de una simulación asumir el papel de un agente y así "pensar como un lobo, una oveja o una mosca" [3].

El desarrollo de Cormas ha tomado una dirección innovadora orientada a la modelización participativa, a saber, el diseño colectivo de modelos y la simulación interactiva. Paralelamente a su desarrollo, se ha puesto en marcha y se ha formalizado una metodología de modelización demonimada ComMod [4], [5]. Este enfoque participativo supone que los actores del desarrollo pueden decidir sus objetivos a largo plazo sobre la base de una comprensión compartida de la situación actual. Entonces es útil explorar escenarios colectivamente para entender mejor si se puede lograr una situación deseada. También es posible interactuar con la simulación para modificar el proceso.

a CIRAD – UPR GREEN, Montpellier, Francia, Bommel, le_page, bousquet, leclerc@cirad.fr
b Universidad de Costa Rica, CIEDA, Costa Rica
c CNRS – UMR 7266 LIENSs, La Rochelle, Francia, nicolas.becu@univ-lr.fr
d Universidad de Brasilia, CDS, Brásil
e CATIE, PIDEA, Costa Rica

1 En informática el término *framework* se refiere a un entorno de trabajo para el desarrollo de aplicaciones, es decir, es un conjunto estandarizado de conceptos y prácticas para facilitar el desarrollo rápido de aplicaciones compatibles.

Para ello, estamos desarrollando Cormas para (a) facilitar el diseño colectivo y la implementación de SMA, y (b) posibilitar el desarrollo de simulaciones interactivas, para que los actores puedan participar activamente, solos o con otros, en el desarrollo de un escenario.

Información general de Cormas

Como un *framework* que ofrece clase predefinidas (que contienen atributos y métodos que pueden ser reutilizados) y un conjunto de herramientas de visualización, Cormas tiene como objetivo facilitar la implementación de SMA, así como el monitoreo y el análisis de simulaciones. Utiliza *VisualWorks* [6], un entorno de programación basado en *Smalltalk*, un lenguaje puramente orientado por objetos (a diferencia de Java o C++ que son una capa al lenguaje C). Existen principalmente tres tipos de entidades genéricas: "agente social", "entidad espacial" y "entidad pasiva". Por lo tanto, para crear un *Ruminante*, por ejemplo, la modelización puede definirlo como un subtipo de *AgentLocation* (agente social localizado). Este agente podrá moverse por el espacio o percibir a sus vecinos llamando a los métodos genéricos (#moveTo: y #perceive:), definidos en la clase *AgentLocation*.

La ventaja de utilizar un *framework* es que libera al modelizador de muchas restricciones de codificación. Esto le permite concentrarse únicamente en su tema sin preocuparse por los accesorios que vienen con un simulador (visualización de espacio, gráficas de evolución temporal, etc.). Después de codificar los agentes y otras entidades, el modelizador debe simplemente indicar cómo ellos son activados por el planificador de simulaciones de Cormas. Finalmente, se puede especificar cómo se quieren visualizar las entidades y las salidas del modelo. Para esta fase existen varias interfaces que evitan codificar las visualizaciones del modelo. Una cuadrícula espacial muestra el paisaje virtual y los agentes, y se pueden seleccionar varias formas de ver el SMA, o sea, según diferentes "puntos de vista" (PoV).

Finalmente, el *framework* permite lanzar baterías de simulación. Recientemente, una conexión con R, un *software* de cálculo estadístico, hace posible realizar análisis de sensibilidad complejos y obtener gráficos estadísticos de calidad.

Modelización participativa con el enfoque ComMod

En paralelo con el desarrollo de Cormas, el enfoque ComMod (para *Companion-Modeling*, o modelización de acompañamiento [5], [7], [8]) se ha desarrollado y aplicado en muchos países. El objetivo es ayudar a los actores del desarrollo a definir sus propios objetivos a largo plazo y "acompañarlos", en lugar de proponer una solución "llave en mano" para la gestión de los recursos naturales (GNR) [9]. En este campo de la GRN parece necesario alejarse de una postura positivista que concibe el conocimiento científico como el único válido. Como en el caso de la epistemología constructivista (en el sentido de [10] y [11]), el enfoque ComMod busca colectivamente "construir" el conocimiento sobre la base de las percepciones y experiencias de las partes interesadas. Más allá de los sistemas tradicionales de apoyo a las decisiones, este enfoque participativo busca construir una representación compartida basada en cómo los actores perciben la situación actual y cómo podría o debería evolucionar. Como método de mediación, ComMod presupone que los actores deben estar bien informados sobre los temas y todos tienen interés en resolver el problema inicial.

Para comprender mejor una situación dada se realiza la concepción colectiva de un SMA para garantizar el reconocimiento de esta representación por todos los actores. Para facilitar esta fase difícil, a menudo se utilizan escenarios donde cada participante desempeña su propio papel en una situación virtual simplificada. Este juego ya es una representación del mundo, por lo tanto un modelo [12]. Al interpretar el papel que representa su actividad en la vida real, un jugador proporciona información para la modelización [13], [14]. Este enfoque se conoce bien en el entorno del llamado "SMA empírico" [15]: las reglas, los comportamientos y las decisiones tomadas durante el juego por los jugadores se utilizan para especificar los agentes del SMA [16]. Las discusiones de lo ocurrido durante el juego que se dan una vez finalizado este, permiten confirmar y/o revisar partes del modelo.

El diseño de un SMA no da acceso inmediato al entendimiento del comportamiento del sistema socioecológico. El tiempo desempeña un papel activo y decisivo en la activación de entidades, y la secuencia de actividades e interacciones produce resultados difíciles de predecir. Incluso si los mecanismos básicos son simples, no somos capaces de tener en cuenta muchos elementos que se influyen mutuamente [17]. Es este escenario en movimiento, esta animación (del latín *animare* "dar vida") que permite al modelo expresarse y explorar escenarios.

En el dominio tradicional de la simulación, los agentes realizan actividades predefinidas y el usuario solo puede observar la evolución del sistema. Ahora Cormas permite interactuar con simulaciones mientras juega con ellas. Estas simulaciones mezclan las decisiones tomadas por los jugadores y otros por el modelo SMA, es decir, permiten cambiar el comportamiento de los agentes y cómo utilizan los recursos durante las simulaciones. El término "modelo de simulación interactivo" que es menos ambiguo.

Nuevas funcionalidades de Cormas, orientadas a la modelización interactiva

En lugar de observar una simulación sin interferir con el proceso, una simulación interactiva tiene como objetivo evaluar las diferentes decisiones tomadas por los agentes. Para ello, los participantes pueden modificar los parámetros o pueden enviar órdenes específicas – o incluso modificar la estrategia principal de un agente. Al interactuar con el sistema virtual a través de su avatar (el agente que juega un rol en el SMA), los participantes pueden probar estrategias alternativas o nuevas prácticas y evaluar sus consecuencias.

Los desarrollos recientes en Cormas pretenden proponer varias herramientas de interacción para facilitar este enfoque interactivo y orientado hacia el futuro, donde los participantes contribuyen activamente al diseño de un SMA e interactúan con el simulador. Estas herramientas han sido desarrolladas usando experimentos concretos de ComMod en el pasado, y se están realizando nuevos progresos para atender las crecientes necesidades de nuevos estudios de casos.

Diseño de un modelo SMA

Como Cormas es un *framework*, el modelizador debe especializar ciertas clases predefinidas, principalmente entidades "sociales", "espaciales" o "pasivas". Al definir una clase, se pueden agregar atributos específicos y especificar sus valores iniciales a través de una tabla. Por ejemplo, el valor inicial del atributo "Energía" de la clase "Ruminante" se puede establecer por defecto en 50 puntos; esto significa que al iniciar una simulación, todos los ruminantes comienzan con 50 puntos de energía, y durante la simulación, cada nueva instancia de Ruminante también comenzará con 50 puntos. De igual forma es posible almacenar los resultados de las simulaciones para analizar el efecto de los valores de los parámetros.

Múltiples ventanas

Con Cormas es posible visualizar el entorno espacial a través de varias ventanas y niveles de Zoom (Figura 1). Debido a que el modelo es independiente de cómo se puede visualizar, se pueden definir y seleccionar diferentes puntos de vista (PoV) para mostrar (o no) las entidades del SMA. En el menú PoV de una cuadrícula espacial se dispone de los PoV para cada clase del modelo. Determinar los PoV es fácil gracias a la interfaz PoVSetter, la cual permite importar imágenes o dibujar figuras correspondientes a diferentes estados de un agente.

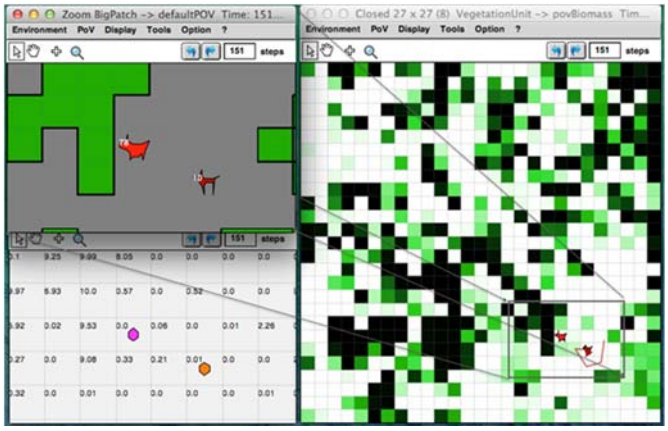

Fig. 1. Cuadrícula espacial completa (derecha) y dos zooms con diferentes puntos de vista (izquierda). Un agente es rastreado en la cuadrícula derecha. Su energía se muestra en la cuadrícula en la parte superior izquierda.

Manipulación de los agentes

Para interactuar con una simulación también es posible actuar directamente sobre el espacio y los agentes que están ubicados en el mismo. Hay dos maneras de actuar, ya sea que se realice en todas las entidades simultáneamente, o bien, solamente en algunas de ellas. En el primer caso podemos cambiar el estado de un grupo de agentes o crear nuevos agentes (Figuras 2a y 2b). En el segundo caso, la herramienta "Manipulación" permite controlar un agente individualmente para mover o enviar mensajes. Un clic derecho en este agente abre un menú contextual que le permite seleccionar un mensaje de una lista que contiene los métodos del agente (Figura 2c).

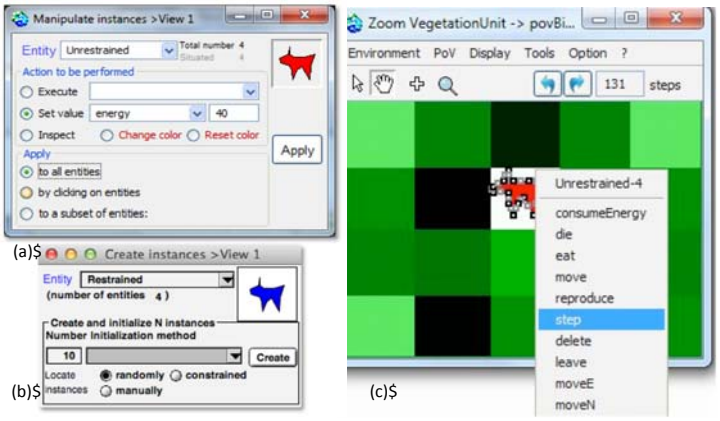

Fig. 2. Interfaces de manipulación: (a) ejecutar una acción o cambiar el valor de los atributos de los agentes, (b) crear nuevas instancias; (c) mover un agente o hacerle ejecutar un método.

Distribución de vistas para la simulación interactiva

En Cormas una simulación puede ser distribuida en varias computadoras. Esto significa que varios usuarios pueden ver de forma remota la misma simulación (desde diferentes puntos de vista) y manipular los agentes. Una simulación se puede distribuir a través del Internet, pero preferimos utilizar esta capacidad en computadoras conectadas en red en una misma sala. En nuestra opinión, la proximidad física es importante porque permite una interacción más rica y más natural a través del diálogo directo o comunicación no verbal.

La simulación con Cormas no se duplica en cada ordenador, sino que solo se distribuyen las vistas y controladores (en acuerdo con la arquitectura Modelo-Vista-Controlador). Así, un solo servidor aloja la simulación y las máquinas cliente muestran puntos de vista específicos y proponen un control de las entidades. Varios usuarios pueden manipular sus agentes y actuar colectivamente en el mismo entorno virtual.

Diagrama de actividad ejecutable

En Cormas un editor permite dibujar diagramas de actividad sencillos. Durante una simulación, estos diagramas son interpretados directamente por los agentes "sobre la marcha", sin tener que traducirlos en código. Por lo tanto, es posible modificar el patrón de comportamiento de un agente sin tener que codificarlo. También puede modificar el simulador en ejecución, sin detener o reiniciar la simulación.

Para simplificar se reducen los elementos disponibles en el editor: nodo inicial y final, puntos de decisión, actividades simples (sin parámetros de entrada o de salida). Al seleccionar una actividad o un punto de decisión en la barra de herramientas, el usuario puede agregar un nuevo elemento al diagrama de actividad. Luego, se debe elegir la operación a realizar por este elemento: esta selecciona en una lista de actividades generada automáticamente a partir de las operaciones disponibles por la clase de destino. Entonces, el usuario puede dibujar una transición entre dos nodos o transiciones verdad/falso desde un punto de decisión. Así, partiendo de las operaciones básicas ya definidas por el modelizador, se puede generar un nuevo comportamiento de nivel superior, sin ningún conocimiento en programación.

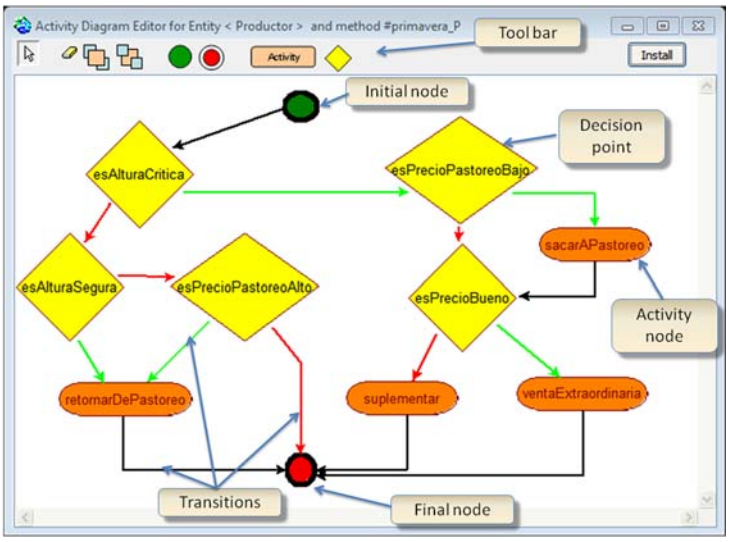

Fig. 3. El Editor de Diagramas de Actividades

El editor no obedece al estándar *Executable UML* (xml) [19], [20] cuyas especificaciones requieren de un compilador para traducir un diagrama en código. En Cormas un diagrama de actividad es interpretado directamente por los agentes. En otras palabras, cada diagrama se guarda en el modelo de la misma manera que el resto del código fuente. Puede ser reabierto en cualquier momento, modificado y ejecutado sin compilación. Aprovechando las ventajas de *Smalltalk*, el cual es un lenguaje de programación reflexivo, es posible modificar el diagrama de un agente mientras se ejecuta la simulación. Tan pronto como se registra, el agente comienza a ejecutar su nuevo comportamiento. Esta especificidad puede ser útil cuando un usuario que observa la tendencia de una simulación quiere probar un cambio en el comportamiento para modificar el curso de la simulación.

El viaje a través del tiempo

Como muchas plataformas, el tiempo avanza por paso de tiempo en Cormas. Ahora es posible volver en el pasado en una simulación. Como el cálculo inverso de las actividades es matemáticamente imposible, Cormas no simula el retroceso en el tiempo. Para activar la función de retroceso, deben estar registrados estados sucesivos del sistema. Por lo tanto, al hacer clic en el botón *Backward*, simplemente se activa la restauración del estado anterior almacenado. Por lo tanto, es posible avanzar o retroceder en un momento particular en el tiempo de simulación, a través de la recarga de un estado grabado.

Esta navegación en el tiempo permite analizar el modelo y comprobar la coherencia de los mecanismos. Al tratar de entender un comportamiento considerado extraño a un punto de la simulación, el usuario puede volver a un momento específico, justo antes de dicho período. Al igual que con una película, se vuelve a ver el proceso para entender cómo y por qué las entidades actúan de tal manera. A partir de este estado en particular, también se puede reiniciar una nueva simulación para comprobar si los agentes se comportan de manera similar o si el sistema evoluciona de manera diferente (bifurcación, ver Figura 4).

Se puede guardar un estado del sistema (instantánea) para convertirse en el punto de partida para futuras simulaciones. Los botones *Undo* y *Redo* de la cuadrícula espacial permiten cancelar una acción del usuario o reactivarla.

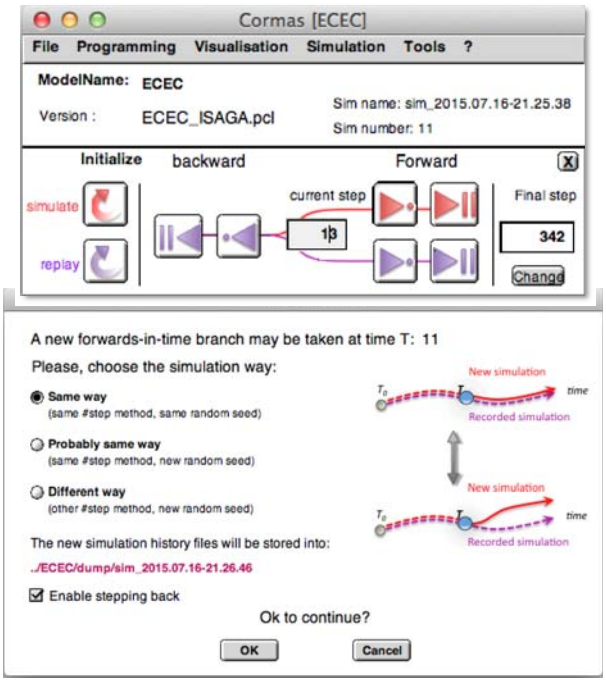

Fig. 4. Arriba: Interfaz principal de Cormas con los botones *Simulate* (rojo) y *Replay* (púrpura). Abajo: Interfaz de bifurcación para iniciar desde un estado grabado

Perspectivas y conclusiones

Cormas está destinado principalmente a los científicos o técnicos no informáticos. El objetivo es facilitar el diseño, la implementación y la evaluación de modelos confiables y eficientes. Para ello, Cormas debe permanecer lo más simple posible. La experiencia ha demostrado que los conceptos generales propuestos por Cormas son adecuados cuando la solución a un problema se basa más en una comprensión colectiva del sistema, que en la precisión temporal de las interacciones.

Para ayudar a los usuarios a implementar sus modelos sin depender de los informáticos, el desarrollo de Cormas se centra en el diseño de interfaces que permiten generar automáticamente parte del código computacional. Además de los diagramas de actividad ya presentes, se encuentra en fase de prueba un editor de diagrama de clases que genera el código de clases, atributos y asociaciones.

Pero el foco principal de Cormas sigue siendo la interactividad con los actores del desarrollo. Es por eso que el esfuerzo principal se dedica a las interfaces hombre-máquina y el diseño ergonómico. Por ejemplo, actualmente estamos desarrollando una extensión para controlar el movimiento de agentes en la cuadrícula espacial a través de objetos reales (peones, tarjetas, etc.) que se ubican físicamente sobre una mesa. Al proyectar la cuadrícula espacial en una tabla, esta extensión se utilizará en simulaciones interactivas combinando entornos virtuales y objetos reales. También estamos desarrollando una mejor visualización del entorno espacial, a través del uso de imágenes (p. e. fondo de mapa o micro-imágenes) o de una conexión con *Google Maps*. Todos estos desarrollos se llevan a cabo en paralelo con experiencias de campo concretas que involucran a varios actores locales con el fin de hacer contribuciones significativas en temas sociales y ambientales complejos.

Agradecimientos

Este documento fue elaborado con el apoyo del programa de investigación FuturAgua financiado por el Foro de Belmont y la Agencia Nacional de Investigación de Francia (ANR). El contenido de este documento es responsabilidad exclusiva de los autores y no puede en modo alguno ser considerado como reflejo de la posición del Foro Belmont. Los autores asumen la responsabilidad colectiva por la calidad del trabajo presentado y publicado.

Reconocimientos

Esta investigación fue conducida como parte del proyecto FuturAgua establecido a través del Fondo Belmont Forum (http://www.belmontforum.org/) y con el apoyo de la Agencia Nacional de Investigación de Francia (ANR). El contenido de este documento es responsabilidad exclusiva de los autores y no puede en modo alguno ser considerado como reflejo de la posición del Foro Belmont. Los autores asumen la responsabilidad colectiva por la calidad del trabajo presentado y publicado.

Implicaciones para la seguridad hídrica

- Las cuestiones relacionadas con el agua son sumamente complejas y es preciso buscar soluciones mediante la participación de todas las partes interesadas. Se supone que los modelos expertos ayudan a encontrar soluciones, pero la experiencia demuestra que este no es el caso para problemas complejos.
- La modelización interactiva facilita el reconocimiento mutuo de las percepciones, la apropiación del conocimiento y la toma de decisiones colectivas. Los Sistemas Multi-Agentes (SMA) son particularmente adaptados para este tipo de modelización.
- Cormas (cormas.cirad.fr) es una plataforma de modelización SMA de acceso libre que se ha demostrado para la gestión adaptativa del agua. Las nuevas funcionalidades de Cormas permiten la participación directa de los actores en el diseño del modelo y durante el transcurso de las simulaciones.

Referencias

[1] Bousquet, F., Bakam, I., Proton, H., Page, C.L. (1998). Cormas : Common-pool resources and multi-agent systems. In Task and Methods in Applied Artificial Intelligences. Del Pobil, A.P. Mira, J., Ali, M. (eds.). Springer Berling Heidelberta, 826-837.

[2] Le Page, C., Becu, N., Bommel, P., Bousquet, F. (2012). Participatory Agent-Based Simulation for Renewable Resource Management: The Role of the Cormas Simulation Platform to Nurture a Community of Practice. Journal of Artificial Societies & Social Simulation, 15(1):10 http://jass.soc.surrey.ac.uk/15/1/10.html

[3] Wilensky, U., Reisman, K. (2006). Thinking like a Wolf, a Sheep, or a Firefly: Learning Biology through Constructing and Testing Computational Theories-An Embodied Modeling Approach, Cognition and Instruction, 24(2):171-209.

[4] Barreteau, O., Antona, M., D'Aquino, P., Abert, S., Boissau, S., Bouquet, F., Daré, W., Etienne, M., Le Page, C., Mathevet, R., Trébuil, G., Weber, J. (2003). Our Companion Modelling Approach. Journal of Artificial Societies and Social Simulation, 6(1):s.p.

[5] Étienne, M. (ed). (2014). Companion Modelling. A Participatory Approach to Support Sustainable Development. Springer Netherlands.

[6] Bauer, J. (2015). The VisualWorks Development Environment. In Programming Smalltalk – Object-Orientation from the Beginning. Springer Fachmedien Wiesbaden. P. 77-96.

[7] Bousquet, F., Barreteau, O., Mullon, C., Weber, J. (1996). Modélisation d'Accompagnement : Systèmes Multi-Agents et Gestion des Ressources Renouvelables. Actas de Coloquio Internacional Quel environment au 21ème siècle? Environment, maîtrise du long terme et démocratie. Abbaye de Fontevraud, Francia.

[8] Gurung, T.R., Bousquet, F., Trébuil, G. (2006). Companion modeling, conflict resolution, and institution building: sharing irrigation water in the Lingmuteychu Watershed, Bhutan. Ecology and Society 11(2):36.

[9] Bousquet, F. Trébuil, G., Hardy, B. (eds.) (2005). Companion Modeling and Multi-Agent Systems for Integrated Natural Resource Management in Asia. International Rice Research Institute. 360 p.

[10] E. von Glasersfeld. (1999). Le Moigne's Defense of Constructivism. Grasce (Ed.). In. Entre systémique et complexité de faisant [Between systemics and complixity – making the way]. Paris, Francia. Disponible en http://www.vonglasersfeld.com/225

[11] Le Moigne, J.L. (1995). Les épistémologies constructivistes. Presses Universitaires de France.

[12] Daré, W., Barreteau, O. (2003). A role-playing gamein irrigated system negotiation : between play and reality. Journal of Artificial Societies and Social Simulation. 6(3): n.p. Disponible en http://jasss.soc.surrey.ac.uk/6/3/6.html

[13] Barreteau, O. (2003). The joint use of role-playing games and models regarding negotiation processes: characterization of associations. Journal of Artificial Societies and Social Simulation, 6(2): n.p. Disponible en http://jasss.soc.surrey.ac.uk/6/2/3.html

[14] Bousquet, F., Barreteau, P., d'Aquino, M., Etienne, S., Boissau, Aubert, S., Le Page, C., Babin, D., Castella, J.C. (2002). Multi-agent systems and role games: collective learning processes for ecosystem management. In. Janssen Marco A. (ed.). Complexity and Ecosystem Management. The Theory and Practice of Multi-Agent Systems. Cheltenham: E. Elgar, p. 249-285.

[15] Janssen, M.A., Ostrom, E. (eds.). (2006). Empirically based, agent-based models. Ecology and Society, 11(2): 37.

[16] D'Aquino, P., Le Page, C., Bousquet, F., Bah, A. (2003). Using Self-Designed Role-Playing Games and a Multi-Agent System to Empower a Local Decision-Making Process for Land Use Management: The SelfCormas Experiment in Senegal, Journal of Artificial Societies and Social Simulation. Disponible en http://jasss.soc.surrey.ac.uk/6/3/5.html

[17] Deffuant, G., Weisbuch, G., Amblard, F., Faure, T. (2003). Simple is beautiful? And necessary. Journal of Artificial Societies and Social Simulation, Disponible en http://jasss.soc.surrey.ac.uk/6/1/6.html

[18] Le Page, C., Abrami, G., Barreteau, O., Becu, N., Bommel, P., Botta, A., Dray, A., Monteil, C., Souchère, V. (2014). Models for sharing representations, in Companion Modelling, Springer Netherlands, p.69-101

[19] Mellor, S.J., Balcer, M. (2002). Executable UML: A Foundation for Model-Driven Architectures. Boston, MA, EE.UU. Addison-Wesley Longman Publishing Co., Inc.

[20] OMG (2008). Model Driven Architecture. MDA Guide Version 1.0.1

Las notas de investigación de FuturAgua sintetizan para la región los resultados científicos del proyecto, que busca aumentar la resiliencia a la sequía de las poblaciones de Guanacaste, Costa Rica.

www.futuragua.ca

Editores: Marianela Argüello L.y Grégoire Leclerc
Traducción: Grégoire Leclerc
Montaje del texto: Marianela Argüello L.
Diseño: Rocío Jiménez, Oficina de Comunicación, CATIE