Simulación Multiagente

Jacinto Dávila
-jacinto@ula.ve-

May 8, 2005

PARTE 1

Agentlink: La Red de Investigación en Tecnología de Agentes en Europa. Un mapa de la ruta.

—http://www.agentlink.org/—

Traducción libre de Jacinto Dávila.

¿Qué es la tecnología de Agentes?

Los agentes pueden ser definidos como entidades computacionales, autónomas, solucionadoras de problemas y capaces de operar efectivamente en ambientes dinámicos y abiertos. A los agentes se les suele implantar en ambientes en los que pueden interactuar, y algunas veces cooperar, con otros agentes (en los que se incluye tanto a software como a personas) que bien pueden tener objetivos que están en conflicto. Esos ambientes son llamados sistemas multiagentes.

A los agentes se les puede distinguir de los objetos (en el sentido de la programación orientada a objetos) puesto que son entidades autónomas, capaces de decidir acerca de sus acciones e interacciones, y puede que actúen para alcanzar objetivos individuales. Son capaces de reinvindicar su autonomía eligiendo como realizar las tareas que se le han asignado o decidiendo sobre las tareas operativas que son necesarias para satisfacer los objetivos de sus usuarios.

Pero lo que es más importante, los agentes toman esas decisiones en el contexto de un ambiente cambiante en el que se encuentran imbuídos. Los agentes no pueden, por consiguiente, se invocados como se hace con los objetos, sino que reciben tareas asignadas por sus propietarios, creadores o usuarios.

Sin embargo, los SMA pueden ser construídos usando una gran variedad de tecnologías, que incluyen a la orientación por objetos, servicios Web y otras. Por su parte, las nociones de agentes pueden ser aplicadas sobre diversos aspectos, algunos de los cuáles se consideran a continuación.

Los agentes como metáfora de diseño

Los agentes se ofrecen a los diseñadores y desarrolladores como una forma de estructurar una aplicación en torno a componentes autónomos y comunicativos, y esto ha llevado a la construcción de herramientas de software e infraestructura computacional para apoyar la metáfora de diseño.

En este sentido, los agentes son también una nueva y también más apropiada ruta para el desarrollo de sistemas complejos, especialmente en ambientes abiertos y dinámicos.

Para soportar esta visión del desarrollo de sistemas, hacen falta herramientas y técnicas particulares. Por ejemplo, metodologías para guiar el análisis y el diseño, arquitecturas de agente para el diseño de componentes individuales, abstracciones para permitir a los desarrolladores lidiar con la complejidad de los sistemas implementados, e infraestructura de soporte (incluyendo otras tecnologías novedosas, como la de servicios Web) que también debe ser integrada.

Los agentes como fuentes de tecnología

Las tecnologías de agentes cumbren un amplio rango de técnicas y algoritmos específicos para lidiar con las interacciones en ambientes abiertos y dinámicos. Aquí se incluyen cosas como el equilibrio entre reacción (reactividad) y el razonamiento en la arquitectura de cada agente, aprender de y sobre otros agentes en el ambiente, descubrir y responder a las preferencias de los usuarios, encontrar formas de negociar y cooperar con otros agentes y desarrollar medios apropiados para formar y liderar coaliciones.

Más aún, la adopción de los enfoques basados en agentes es cada vez más importante en otros dominios. Por ejemplo, los sistemas multiagentes ya proveen métodos más eficientes y efectivos para asignación de recursos en ambientes complejos, tales como la gestión de redes de utilidad, que ningún método centrado en humanos.

Los Agentes como una simulación

Los sistemas multi-agente ofrecen poderosos modelos para representar ambientes realistas con un grado apropiado de complejidad y dinamismo. Por ejemplo, la simulación de economías, de sociedades o de ambientes biológicos son áreas típicas de aplicación.

El uso de sistemas de agentes para simular dominios reales puede ofrecer respuestas a problemas complejos, tanto físicos como sociales, que no podrían ser obtenidas de otra manera, tales como en el modelado del impacto de los cambios climáticos sobre las poblaciones biológicas, o el modelado del impacto de las alternativas de políticas públicas en la conducta social o económica.

La simulación basada en agentes cubre:

estructuras e instituciones sociales que puedan explicar razonablemente los fenómenos observados, que puedan ayudar en el diseño de estructuras organizacionales, y que puedan servir para informar a quienes toman decisiones gerenciales o políticas;

sistemas físicos, tales como edificios inteligentes, sistemas de tráfico y poblaciones biológicas; y

sistemas de software de todos los tipos, incluyendo el comercio electrónico y las agencias de información.

Los sistemas de agente no son panaceas para todos esos grandes problemas. Pero han demostrado ofrecer ventajas competitivas concretas tales como:

- Mejorar la robustez operativa con recuperación inteligente frente a fallas;

- Reducir los costos de las provisiones al computar las políticas más beneficiosas para adquisiciones en mercados en línea; y

- Mejorar la eficiencia de los procesos de manufactura en ambientes dinámicos.

¿Qué haremos en este curso?

Un ejercicio controlado de modelado y simulación multi-agente.

¿Cómo lo vamos a hacer?

Todo construido en un ambiente cooperativo y producido como software libre GNU.

Coloque al comienzo de cada archivo fuente el nombre del programa o módulo y una breve descripción de lo que hace. Copyright (C)

```
This program is free software; you can redistribute it and/or modify it under
the terms of the GNU General Public License as published by the Free Software
Foundation; either version 2 of the License, or (at your option) any later
version.

This program is distributed in the hope that it will be useful, but WITHOUT
ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS
FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with
this program; if not, write to the Free Software Foundation, Inc., 59 Temple
Place, Suite 330, Boston, MA 02111-1307 USA
```

Agregue, al final de ese encabezado que declara la licencia, sus datos completos incluyendo su email permanente.

Más detalles:

—http://www.opensource.org/licenses/gpl-license.php—

Pero . . . ¿Qué, exactamente, vamos a hacer?

Tendencias emergentes y tendencias propulsoras críticas (según —www.agentlink.org—).

Servicios Web Medios standard para la interoperabilidad entre aplicaciones de software que operan en diversas plataformas ...

Computación Grid La "Grid" es LA infraestructura de computación de alto rendimiento para soportar proyectos científicos distribuidos y de gran escala ...

Inteligencia Ambiental La noción de Inteligencia Ambiental surge principalmente gracias a los esfuerzos de la comisión europea para identificar los desafíos en Europa en cuanto a las tecnologías para la sociedad de la información... (Provisión "transparente" de servicios y aplicaciones)

Computación Autonómica Propuesto originalmente por IBM, se trata de un método para la autogerencia de sistemas de cómputo con un mínimo de intervención humana. (incrementar la productividad y minimizar la complejidad para los usuarios finales).

La Web Semántica Está basada en la idea de que los datos en la Web (WWW) pueden ser definidos y enlazados de tal forma que puedan ser procesados por máquinas para ser procesados automáticamente e integrados para diversos propósitos. (La Web actual está limitada por la habilidad de los humanos para los datos en las diversas fuentes de información).

Sistemas Complejos Los sistemas de agentes proveen un medio para conceptualizar a los SC y luego simularlos.

Pero ... ¿Qué, exactamente?

Problemas propuestos por el Prof. Carlos Domingo para el diseño de su **Sociedad Flexible**:

—http://webdelprofesor.ula.ve/economia/carlosd/—

- Simulación de la Ley de Ashby.
- Enseñanza isocéntrica. El agente help.
- Enseñanza por descubrimientos.

- Dinámica de los grafos.
- Formación y limitación de Jerarquías.
- Redes de empresas en colaboración y competencia.
- Generación de mecanismos top-down.
- Modelado de Revoluciones.
- El tamaño máximo de los grupos.
- Problemas de depósitos de datos.

PARTE 2

El Agente: ¿Cómo programarlo?

Una respuesta basada en la teoría de agentes propuesta por Kowalski.

—http://webdelprofesor.ula.ve/ingenieria/jacinto/kowalski/logica-de-agentes.html—

Programa de Agent 1, Agent 2, Agente 4.

Jacinto Davila <jacinto@ula.ve>. Grupo Galatea. CESIMO. Universidad de Los Andes. Merida. Venezuela

Agent 3 was developed originally by Bob Kowalski

Last modified: May, 2005

```
Agent 1:

Goals: Stay alive

Beliefs: To stay alive
    whenever need-air then breathe and
    whenever need-food then eat and
    whenever need-help then get-help
```

```
Agent 2:

Goals: Stay alive

Beliefs: Somebody stays alive if
        consider breathing and
        consider eating and
        consider protecting-self

        An agent consider breathing if
           need-air and breath
        or do not need-air.

        ...
```

Las metas de un agente calculador

```
Agent 3

Goals: If I am offered a deal and
    it benefits me and
    it hurts somebody more than it benefits me and
    that person is my friend then I turn off the deal

    If I am offered a deal and
    it benefits me and
    it hurts somebody and
    that person is NOT my friend then I accept the deal

    If I am offered a deal and
    it hurts somebody then I turn off the deal.
```

Las metas de un agente hedonista

Agente 4

Meta Superior: Obtener el máximo placer por el mayor tiempo posible.

Creencias: Obtengo el máximo placer durante más tiempo si monopolizo la atención de alguien y le uso para causarme placer y le mantengo en buena condición entre tanto y por más tiempo.

Obtengo el máximo placer durante más tiempo si acumulo mucho dinero y compro placer a buen precio.

Obtengo el máximo placer durante más tiempo si adquiero una posición de privilegios y la disfruto al máximo.

¿Cómo funciona un agente ascensor?

La traza a continuación puede ser resumida así: En el momento *now*, el agente, sabiendo que está en el piso 3 (*floor 3*), observa que el botón en el piso 5 (*floor 5*) ha sido presionado. Esta **entrada** (*input*) activa la acción de subir (*go up*). Este tipo de activación ocurre dos veces y la ejecución de los respectivos "comandos", lleva al elevador al piso 5. Una vez allí, el razonador (al que llamamos **demo**) activa las metas de abrir la puerta (*open door*) y apagar la señal (*turn off*) de llamada.

$mental\ activity$	**demo**
$inputs$	$I : atfloor(3, now), on(5, now)$
$time$	$T : now$
$goals$	$N_1 = \{true\} + \{true\}$

At the beginning the elevator is at floor 3 and the term *now* refers to the time at that moment. Let us assume as above that $atfloor(3, now)$ and $on(5, now)$ (the button at floor 5 is on) are known to the agent. The reasoner module ($demo$) is starting to run.

$\downarrow$

$mental\ activity$	**act**
$inputs$	$I : \ldots$
$time$	$T : now + r_1$
$resources$	$R : r_1$
$goals$	$N_1 = \{do(up, T_1), now < T_1\} + \{\ldots\}$

$demo$ has completed its portion of time this cycle, leaving activated the goals of *going up*, from floor 3, presumably to serve floor 5, at some time T_1 after now. Observe that, because $demo$ took r_1 units of time for reasoning, the new current time is now + r_1. Also note that the store of inputs is unchanged w.r.t previous time and that act is starting its execution.

$mental\ activity$	**demo**
$inputs$	$I' : atfloor(4, now + r_1 + 1),$ $atfloor(3, now), on(5, now)$
$time$	$T : now + r_1 + 1$
$resources$	$R' :?$
$goals$	$N_1 = \{do(up, now + r_1), \ldots\} + \{\ldots\}$

act successfully executed the action of moving up at time $now + r_1$. It took act 1 unit of time to do so, and then the current time is $now + r_1 + 1$. As part of the feedback, the agent has learnt that it is at floor 4 at time $now + r_1 + 1$. $demo$ takes over once again.

$\downarrow$

$mental\ activity$	**act**
$inputs$	$I' : \ldots$
$time$	$T : now + r_1 + 1 + r_2$
$resources$	$R' : r_2$
$goals$	$N_1 = \{do(up, T_2), now + r_1 + 1 < T_2,$
	$do(up, now + r_1), \ldots\} + \{\ldots\}$

When $demo$ suspends its running, time is $now + r_1 + 1 + r + 2$ ($demo$ consumed r_2 unit reasoning) and a new goal, similar to the one above, has been activated. The elevator must go up again, any time T_2 after $now + r_1 + 1$ which is, of course, already in the past. Act re-starts to run.

$\downarrow$

$mental\ activity$	**demo**
$inputs$	$I'' : atfloor(5, now + r_1 + 1 + r_2 + 1),$
	$\quad atfloor(4, now + r_1),$
	$\quad atfloor(3, now), on(5, now)$
$time$	$T : now + r_1 + 1 + r_2 + 1$
$resources$	$R'' :?$
$goals$	$N_1 = \{do(up, now + r_1 + 1 + r_2), \ldots\} + \{\ldots\}$

Once again, act succeeded in executing the corresponding action and has learnt its new position. Time is $now + r_1 + 1 + r_2 + 1$ when $demo$ starts again reasoning.

$\downarrow$

$mental\ activity$	**act**
$inputs$	$I'' : \ldots$
$time$	$T : now + r_1 + 1 + r_2 + 1 + r_3$
$resources$	$R'' : r_3$
$goals$	$N_1 = \{do(open, T_3), do(turnoff, T_3),$ $do(close, T_4), \ldots\} + \{\ldots\}$

$demo$ has suspended its processing, this time after activating a more complex set of goals. Two actions in that set are next for parallel execution at time T_3. With that set of instructions, to start serving floor 5, act resumes its running at time $now + r_1 + 1 + r_2 + 1 + r_3$.

$\downarrow$

Asumimos que en cada llamada a demo, el tiempo que se dedica a computar el qué hacer es suficiente para que todas las opciones sean consideradas. Este, desde luego, no es siempre el caso. Uno tendría que analizar otros patrones de ejecución para evaluar el programa del agente.

PARTE 3

Una teoría de simulación multiagente

... hecha en casa (pero no desde cero).

La teoría de sistemas multiagentes propuesta por Dávila, Uzcátegui y Tucci se apoya en los siguientes conceptos fundamentales:

- Influencias.
- La reacción del ambiente.
- La descomposición del estado global de un sistema dinámico en todos tipos de componentes: influencias y variables del estado global.
- La descripción de las dinámicas de cada elemento del sistema usando máquinas abstractas.

Un agente reactivo y racional

$$< k'_a, g'_a, \gamma'_a >= conducta_a(t, r_a, k_a, g_a, \gamma) \qquad (1)$$

con

t : el tiempo actual.
r_a : la cantidad de tiempo para razonamiento del agente.
k_a : la base de conocimientos del agente.
g_a : el conjunto de metas del agente.
γ : el conjunto de influencias ya en la historia.
γ_a : el conjunto de influencias que postula el agente.

donde $conducta$ hace referencia a

$$k'_a = actualiza_a(t, percibe_a(\gamma), k_a) \quad (2)$$

$$< \gamma'_a, g'_a > = planifica_a(t, r_a, k'_a, g_a) \quad (3)$$

El sistema multi-agente con ese agente racional

Ferber y Müller llamaron $Evolution$ a la función que describe como progresa todo un sistema multiagente al pasar el tiempo. Para mantener la consistencia, aquí la llamaremos $evolucion$. Ésta función puede definirse así:

$$evolucion(t, < s_1, s_2, .., s_n >, \sigma, \gamma) \; = \\ evolucion(paso(< s_1, s_2, .., s_n >, t, \sigma, \gamma)) \tag{4}$$

donde

$$s_a =< k_a, g_a > \tag{5}$$

describe el estado interno del agente a y puede obtenerse a partir de 1.

Nótese que se trata simplemente de una función recursiva que se invoca a sí misma, luego de invocar a la función $paso$ que describe la transición de un estado global al siguiente ($cycle$ para Ferber y Müller).

Desde luego, los detalles interesantes de cómo un estado global se trasforma en otro tomando en cuenta la evolución de los agentes, las influencias que estos postulan y la reacción del ambiente a dicha influencia, tienen que ser incluidos en la definición de $paso$, por tanto

$$paso : S \times \Im \times \Sigma \times \Gamma \rightarrow S \times \Im \times \Sigma \times \Gamma$$

puede ser representada como

$$< t', < s'_1, s'_2, .., s'_n >, \sigma', \gamma' >= paso(< s_1, s_2, .., s_n >, t, \sigma, \gamma) \quad (6)$$

la cual hace referencia a varias funciones, entre ellas a la función $conducta$ de la ecuación (1) para obtener los s_a y los γ_a y la función $reacciona$ definida como

$$< \sigma', \gamma' >= reacciona(\Lambda, \beta, t, \sigma, \gamma \cup_a \gamma_a) \quad (7)$$

donde

t : el tiempo actual.
s_a : el estado interno del agente a.
σ : las variables de estado del sistema.
γ : el conjunto de influencias ya en la historia.
γ_a : el conjunto de influencias que postula el agente a.
Λ : la descripción del sistema a través de las leyes de cambio.
β : la estructura e información de soporte del sistema.

Esta descripción matemática dice, en esencia, que el sistema evoluciona de dos maneras:

(a) por la respuesta del ambiente al estado actual y a las influencias de los agentes, procesadas en $reacciona$ usando descripciones del cómo debe cambiar el sistema (Λ, β).

(b) por el progreso de los agentes en la producción de las influencias ($conducta$).

FIN