# Programación de Agentes y Argumentación

Sebastián Gottifredi  Alejandro J. García  Guillermo Simari
Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET),
Laboratorio de Investigación y Desarrollo de Inteligencia Artificial,
Departamento de Ciencias e Ingeniería de la Computación,
Universidad Nacional del Sur,
Avendia Alem 1253 - (B8000BCP) - Bahía Blanca - Argentina
Tel: (0291) 459-5135 / Fax: (0291) 459-5136
Email: {sg,ajg,grs}@cs.uns.edu.ar

**Resumen**

Esta línea de investigación involucra programación de agentes y argumentación. En particular, en este trabajo se presentan las motivaciones y las investigaciones en curso.

El principal objetivo de esta linea, es el desarrollo de herramientas que permitan una programación declarativa de agentes inteligentes. En especial, agentes que utilizan razonamiento nomonótono y de sentido común. Particularmente, se buscan herramientas que cuenten con mecanismos de argumentación para el razonamiento de los agentes.

Actualmente se están estudiando diferentes arquitecturas de agente, que utilizan argumentación como mecanismo de razonamiento, y lenguajes de programación de agente, que permite implementaciones declarativas. Aprovechando este análisis y como primer paso para permitir que la programación de agente que razonan utilizando argumentación, se han presentado constructores que permitan formar argumentos para garantizar las creencias del agente.

Palabras claves: Argumentación, Lenguaje de Especificación de Agentes, Lenguajes de Programación de Agentes.

## 1. Introducción

La importancia del uso de agentes inteligentes para resolver problemas complejos es bien conocida en la literatura [21]. Este tipo de agentes suelen estar caracterizados a través componentes mentales como creencias, deseos, compromisos o intenciones. Entre estos agentes, los mas comunes, son aquellos que están basados en la teoría BDI (belief, desires, intentions) [3, 16, 12].

Hoy en día, aun son necesarias herramientas que permitan especificar y programar este tipo de agentes en términos de sus componentes mentales de una manera declarativa [13, 8, 9]. En particular, en esta línea de investigación, estamos interesados en herramientas que además de permitir una forma declarativa de especificar los componentes mentales, provean mecanismos de argumentación para el razonamiento.

La idea de usar argumentación como parte del proceso de razonamiento de un agente inteligente no es nueva. Existen varias aproximaciones anteriores a esta línea que relacionan agentes cognitivos

Financiado parcialmente por CONICET (PIP 5050), Universidad Nacional del Sur (PGI 24/ZN11) y Agencia Nacional de Promoción Científica y Tecnológica.

(especialmente aquellos que siguen la teoria BDI) con frameworks de argumentación [1, 14, 15, 17]. Sin embargo, en ninguna de ellas se toman todas las decisiones teóricas y de diseño necesarias para un lenguaje de programación de agente, que en particular utiliza los conceptos de argumentación.

## 1.1. Lenguajes de Programación de Agente

Los lenguajes de programación de agente deben proveer herramientas al usuario para programar y ejecutar agentes. La cuestión clave en este area es determinar la forma correcta de programar agentes. Esto es, establecer claramente que primitivas debe proveer un "lenguaje de programación orientado a agentes" para construir agentes autónomos, interactivos, racionales.

La primer formalización del concepto de lenguaje de programación de agente surge con el paradigma propuesto por Shoham en [18], llamado programación orientada a agentes. En ese trabajo se exponen las características con las que debe contar un lenguaje de programación de agente:

- Un sistema lógico para definir el estado mental del agente.
- Un lenguaje para programar los agente (especificar su código).
- Un proceso de compilación de para convertir a los agentes en sistemas ejecutables de bajo nivel.

Estas ideas han sido a tomadas, refinadas y modificadas por muchos investigadores en búsqueda de "buenos" lenguajes de programación de agente. En particular, en [21, 20] se divide a la programación de agentes en tres areas principales: teorías de agente, arquitecturas de agente y lenguajes de programación de agentes. Las teorías de agente, están relacionadas con los formalismos para utilizados para especificar las propiedades de los agentes. Las arquitecturas de agente, están relacionadas a la construcción de los componentes mentales y sus relaciones, cumpliendo las propiedades de una teoría de agente. Finalmente, los lenguajes de programación de agentes, están relacionados con las primitivas utilizadas para programar e implementar agentes cumpliendo las propiedades de una teoría de agente y/o una arquitectura de agente.

En la actualidad lenguajes de programación de agente como Jason [2] o 3APL [5], toman como base todos estos conceptos. Estos lenguajes de programación de agente utilizan la teoría BDI y permiten al usuario programar agentes a través de sus componentes mentales, de una manera declarativa y sin perder los formalismos lógicos subyacentes. Por ultimo, en la actualidad la mayoría de los lenguajes de programación de agente proporcionan un soporte para implementar (en mayor o menor medida) sistemas multi-agente.

## 1.2. 3APL

En el area de los lenguajes de programación de agente, 3APL [5, 4, 11] es una interesante propuesta, que permite implementar agentes de una manera declarativa a través de la especificación de sus estados mentales. 3PAL fue presentado en [11] y luego extendido en [4, 5]. Este lenguaje sigue el espíritu de la programación orientada a agente [18] y Dribble [19] (un lenguaje de programación de agente basado en metas).

3APL combina programación en lógica (para la especificación de los componentes mentales de los agentes) y programación imperativa (para la implementación de la estructura de los planes). Desde el punto de vista imperativo, 3APL hereda la gran mayoría de los constructores imperativos (if, while, etc.), incluido los procedimientos recursivos y la computación basada en estados. Sin embargo, los estados de un agente en 3APL son bases de creencias, que es distinto de la usual asignación de variables de la programación imperativa. Desde el punto de vista de la lógica computacional, las respuestas para las consultas a la base de creencias son pruebas en el sentido de la programación en lógica.

Este lenguaje provee constructores para implementar creencias, metas, capacidades básicas como actualización de creencias o acciones y un conjunto de reglas de razonamiento que son utilizadas por el agente para actualizar o revisar sus metas. Las creencias de un agente 3APL son especificadas por un programa lógico, y es posible utilizar un conjunto de reglas mentales (capacidades) para modificar dinámicamente este programa. Por otra parte, las reglas de razonamiento relacionan metas con planes, y serán ejecutadas siempre que se cumpla una condición determinada por las creencias. Los programas de 3PAL son ejecutados por un interprete que implementa un ciclo deliberativo utilizando cada uno de los consntructores.

### 1.3. Argumentación en representation de conocimiento y razonamiento

La argumentación es una aproximación para el razonamiento con información inconsistente basado en la construcción y comparación de argumentos [6]. En este tipo de razonamiento se producen argumentos a favor y en contra de alguna conclusion y son evaluados para probar la plausibilidad de la conclusion. Las conclusiones pueden se utilizadas para representar decisiones, deseos, opiniones, etc.

Desde un punto de vista de los agentes, es posible utilizar argumentación para representar sus deseos, creencias e intenciones [1, 14, 15, 17]. Por ejemplo, es posible exponer razones a favor y en contra (argumentos) para determinar si adoptar un deseo (conclusion) o no. El mecanismo de argumentación pesará las razones y determinará si el deseo es adoptable.

Las investigaciones en programación en lógica, rozamiento no monótono y argumentación han llegado a importantes resultados, otorgando poderosas herramientas para la representación de conocimiento y el razonamiento de sentido común. Un ejemplo de estas herramientas es programación en lógica rebatible (DeLP) [7].

## 2. Tareas en Curso

Dadas las ventajas que representa la argumentación como lenguaje de representación de conocimiento y razonamiento [6], y la existencia de herramientas concretas sofisticadas para especificar argumentos [7] resulta interesante incluir razonamiento argumentativo en lenguajes de programación de agente. De esta manera, utilizando estos lenguajes se podrán representar creencias, deseos, compromisos, etc. utilizando reglas declarativas que generaran argumentos y se aprovechara el proceso de argumentación para determinar que cree, desea, compromete, etc. el agente. Todo esto sin dejar de respetar los ideales del paradigma orientado a agentes [21].

Por lo tanto, en esta linea de investigación se encuentran en desarrollo lenguajes de programación de agente que respetan el paradigma orientado a agentes, son altamente declarativos y utilizan argumentación. En particular, hemos utilizado 3APL por su declarativad, combinación lógico imperativa en planificación y su soporte para sistemas multi-agente. En [10] se realizo un estudio de dos propuestas para el desarrollo de agentes, una arquitectura [17] y el lenguaje de programación de agente [5]. A partir de ese estudio, se presento un lenguaje de programación basado en 3APL, llamado 3APL-DeLP, que permite representar las creencias del agente mediante reglas DeLP. Esto le permitirá al agente expresar razones a favor (cuerpo de la regla) de una creencia inferible (cabeza de la regla). A partir de estas reglas se construirán los argumentos. Estos serán utilizados por el proceso de razonamiento argumentativo para determinar entre aquellas creencias contradictorias cual prevalece, según algún criterio de comparación. Dado que 3APL-DeLP es un lenguaje de programación de agente, fue necesario definir claramente la sintaxis y semántica de los constructores para representar las creencias, ya que permiten el uso de lo mecanismo de argumentación en combinación con los viejos constructores. Además, dado que de esta manera se incrementan las capacidades de representación de creencias,

fue necesaria una revision de la interacción de la base conocimiento con los otros componentes mentales del agente. Para ello, se redefinieron adecuadamente las reglas de razonamiento y las acciones mentales, de manera tal que aprovechen los nuevos elementos de alto nivel otorgados por DeLP.

# 3. Trabajo a Futuro

Como trabajo a futuro, se esta estudiando un lenguaje de programación donde los agentes consideran otros elementos del agente, justificados bajo algún criterio, utilizando argumentación. En [17, 1, 15] se presentan diferentes arquitecturas que contemplan estos conceptos, pero en el area de los lenguajes de programación de agentes es necesario tomar decisiones de diseño respecto a los constructores, para considerar su inclusion.

# 4. Conclusiones

En este trabajo se ha presentado la linea de programación de agentes y argumentación. Se han mostrado las motivaciones, las propuestas y el trabajo a futuro de la linea. En particular, se ha mostrado resumen de la propuesta para un lenguaje de programación de agente con constructores que permitan formar argumentos para garantizar las creencias del agente.

Desde un punto de vista de lenguajes de programación de agentes, en nuestra propuesta se ha mejorado la capacidad representacional de las creencias utilizando mecanismo que permite mayor nivel de abstracción. Lo cual, lleva a la implementación de agentes cognitivos mas sofisticados.

A diferencia de las arquitecturas de agente que utilizan argumentación, en nuestra propuesta se da forma a todas aquellas decisiones de diseño no necesarias en una arquitectura de agente, como por ejemplo formas de actualizar la base de creencias, semánticas entre los componentes y formas de instanciación de variables.

# Referencias


[1] Leila Amgoud. A formal framework for handling conflicting desires. In *ECSQARU*, pages 552–563, 2003.

[2] Rafael H. Bordini, Jomi Fred Hübner, and Renata Vieira. Jason and the golden fleece of agent-oriented programming. In *Multi-Agent Programming*, pages 3–37. 2005.

[3] M. E. Bratman, D. Israel, and M. Pollack. Plans and resource-bounded practical reasoning. In *Philosophy and AI: Essays at the Interface*.

[4] M. Dastani, B. van Riemsdijk, F. Dignum, and J. Meyer. A programming language for cognitive agents: Goal-directed 3apl. In *First Workshop on Programming Multiagent Systems: Languages, frameworks, techniques, and tools (ProMAS03), Melbourne, Australia*, 2003.

[5] Mehdi Dastani, M. Birna van Riemsdijk, and John-Jules Ch. Meyer. Programming multi-agent systems in 3apl. In *Multi-Agent Programming*, pages 39–67. 2005.

[6] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.*, 77(2):321–358, 1995.

[7] A. Garcia and G. Simari. Defeasible logic programming: An argumentative approach. *Theory and Practice of Logic Programming*, 4(1-2):95–138, 2004.

[8] S. Gottifredi and A. Garcia. Análisis de lenguajes de implementación de agente. In *Workshop de Investigadores en Ciencias de la Computación (WICC 2006)*, 2006.

[9] Sebastian Gottifredi and Alejandro J. Garcia. Análisis lenguajes de especificación de agente en robótica móvil. In *9vo. Workshop de Investigadores en Ciencias de la Computación (WICC 2007)*, pages 27–31, 2007.

[10] Sebastian Gottifredi, Alejandro J. Garcia, and Guillermo R. Simari. Agent programming using defeasible argumentation for knowledge representation and reasoning. In *13vo. Congreso Argentino de Ciencias de las Computación (CACIC 2007)*, pages 1464–1475, 2007.

[11] K. Hindriks, F. de Boer, W. van der Hoek, and J.-J. Ch. Meyer. Agent programming in 3apl. *Autonomous Agents and Multi-Agent Systems*, 2(4):357–401, 1999.

[12] V. Mascardi, D. Demergasso, and D. Ancona. Languages for programming BDI-style agents: an overview. In F. Corradini, F. De Paoli, E. Merelli, and A. Omicini, editors, *Proceedings of WOA 2005: Dagli Oggetti agli Agenti. 6th AI*IA/TABOO Joint Workshop "From Objects to Agents"*, pages 9–15. Pitagora Editrice Bologna, 2005.

[13] Fisher Michael, Bordini Rafael, H. Hirsch Benjamin, and Torroni Paolo. Computational logics and agents: A road map of current technologies and future trends. *Computational Intelligence*, 23(1):61–91, February 2007.

[14] Simon Parsons, Carles Sierra, and Nicholas R. Jennings. Agents that reason and negotiate by arguing. *J. Log. Comput.*, 8(3):261–292, 1998.

[15] Iyad Rahwan and Leila Amgoud. An argumentation based approach for practical reasoning. In *AAMAS*, 2006.

[16] A. S. Rao and M. P. Georgeff. Modeling rational agents within a BDI-architecture. In James Allen, Richard Fikes, and Erik Sandewall, editors, *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91)*, pages 473–484. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA, 1991.

[17] N. D. Rotstein, A. J. Garcia, and G. R. Simari. Reasoning from desires to intentions: A dialectical framework. In *22nd. AAAI Conference on Artificial Intelligence*, July 2007.

[18] Y. Shoham. Agent-oriented programming. In *Artificial Intelligence*, 1993.

[19] Birna van Riemsdijk, Wiebe van der Hoek, and John-Jules Ch. Meyer. Agent programming in dribble: from beliefs to goals using plans, 2003.

[20] M. Wooldridge. Intelligent agents. In *Multiagent Systems*. 1999.

[21] Michael Wooldridge and Nicholas R. Jennings. Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10(2):115–152, 1995.