

## References

- ALÓ, C.C. **Uma Abordagem para Transparência em Processos Organizacionais Utilizando Aspectos**. Tese de Doutorado, Pontifícia Universidade Católica do Rio de Janeiro, 328 páginas, Agosto 2009.
- AMYOT, D.; GHANAVATI, S.; HORKOFF, J.; MUSSBACHER, G.; PEYTON, L.; YU, E. **Evaluating goal models within the goal-oriented requirement language**. *International Journal Intelligent Systems* 25, pp. 841-877, August 2010.
- ASNAR, Y. and GIORGINI, P. **Modeling Risk and Identifying Countermeasure in Organizations**. *International Workshop on Critical Information Infrastructures Security (CRITIS'06)*, Springer, LNCS, 4347, pp. 55-56, 2006.
- ASNAR, Y.; BRYL, V.; GIORGINI, P. **Using Risk to Evaluate Design Alternatives**. *Agent-Oriented Software Engineering (AOSE VII)*, Springer, LNCS, 44405, pp. 140-155, 2007.
- BASILI, V.R. **Software Modeling and Measurement: The Goal Question Metric Paradigm**. *Computer Science Technical Report Series, CS-TR-2956 (UMIACS-TR-92-96)*, University of Maryland, College Park, MD, September 1992.
- BAUER, B.; MÜLLER, J.P.; ODELL, J. **Agent UML: A Formalism for Specifying Multiagent Software Systems**. *Agent-Oriented Software Engineering, LNCS, Volume 1957/2001*, 109-120, DOI: 10.1007/3-540-44564-1\_6, 2001.
- BELLIFEMINE, F.; CAIRE, G.; GREENWOOD, D. **Developing Multi-Agent Systems with JADE**. John Wiley & Sons, ISBN 0470057475, 286 pages, April 2007.
- BERRY, D.M.; CZARNECKI, K.; ANTKIEWICZ, M.; ABDELRAZIK, M. **Requirements Determination is Unstoppable: An Experience Report**. *17th IEEE Int. Requirements Eng. Conference*, pp. 311-316, 2010.
- BERTOLINI, D.; DELPERO, L.; MYLOPOULOS, J.; NOVIKAU, A.; ORLER, A.; PENSERINI, L.; PERINI, A.; SUSI, A.; TOMASI, B. **A tropos model-driven development environment**. In: Boudjlida, N.; Cheng, D.; Guelfi, N. (eds.) *CAiSE Forum, CEUR Workshop Proceedings*, vol. 231, 2006.

- BRATMAN, M. E. **Intention, Plans, and Practical Reason**. University of Chicago, ISBN: 1575861925, 208 pages, 1999.
- BRAUBACH, L., POKAHR, A.; LAMERSDORF, W. **Jadex: A Short Overview**. Net. ObjectDays'04: AgentExpo, pp. 195-207, September 2004.
- BRAUBACH, L.; LAMERSDORF, W.; POKAHR, A. **Jadex: Implementing a BDI Infrastructure for JADE Agents**. Distributed Systems and Information Systems, vol. 3, n. 3, pp.76-85, September 2003.
- BRAUBACH, L.; POKAHR, A.; LAMERSDORF, W. **Extending the Capability Concept for Flexible BDI Agent Modularization**. 3rd Int. Workshop on Programming Multi-Agent Systems (ProMAS'05), pp.99-114, 2005.
- BRESCIANI, P.; PERINI, A.; GIORGINI, P.; GIUNC, F.; MYLOPOULOS, J. **Tropos: An Agent-Oriented Software Development Methodology**. Autonomous Agents and Multi-Agent Systems, Kluwer Academic Publishers, 8, pp. 203–236, 2004.
- BRIN, S. and PAGE, L. **The Anatomy of a Large-Scale Hypertextual Web Search Engine**. Computer Networks and ISDN Systems, Elsevier Science Publishers, pp. 107-117, 1998.
- BROOKS, R.A. **Intelligence without Reason**. 12th International Joint Conference on Artificial Intelligence, pp. 569-595, 1991.
- BRYL, V.; GIORGINI, P.; MYLOPOULOS, J. **Supporting Requirements Analysis in Tropos: a Planning-Based Approach**. Pacific Rim International Workshop on Multi-Agents (PRIMA'07), Springer, 5044, pp. 243-254, 2007.
- BRYL, V.; MASSACCI, F.; MYLOPOULOS, J.; ZANNONE, N. **Designing Security Requirements Models through Planning**. 18th International Conference on Advanced Information Systems Engineering (CAiSE'06), Springer, LNCS, 4001, pp. 33-47, 2006.
- BUSETTA, P.; RÖNNQUIST, R.; HODGSON, A.; LUCAS, A. JACK **Intelligent Agents – Components for Intelligent Agents in Java**. Technical Report TR9901, AOS, January 1999. Accessed at: <http://www.jackagents.com/pdf/tr9901.pdf> (Jan 2011)
- C&L. **Ferramenta Cenários e Léxicos**. Disponível em: <http://pes.inf.puc-rio.br/cel/> (último acesso em Junho 2011)

- CARTER, R.A.; ANTÓN, A.I.; WILLIAMS, L.; DAGNINO, A. **Evolving Beyond Requirements Creep: A Risk-Based Evolutionary Prototyping Model**. Fifth IEEE International Symposium on Requirements Engineering (RE'01), pp. 94, 2001.
- CASTOR, A.; PINTO, R.; CASTRO, J.; SILVA, C. **Towards Requirement Traceability in TROPOS**. VII Workshop on Requirements Engineering (WER'04), Tandil, Argentina, 2004.
- CASTOR, A.P. 2004. **Rastreamento de Requisitos no processo de desenvolvimento de software orientado a agentes**. Dissertação de Metrado. Universidade Federal de Pernambuco, Departamento de Ciência da Computação do Centro de Informática, Recife, Brasil, Agosto 2004.
- CASTRO, J.; KOLP, M.; MYLOPOULOS, J. **Towards requirements-driven information systems engineering: the Tropos project**. 13th International Conference on Advanced Information Systems Engineering (CAiSE\*01), ISSN:0306-4379, Vol. 27, Issue 6, pages 365-389, September 2002.
- CASTRO, J.; PINTO, R.; CASTOR, A.; MYLOPOULOS, J. **Requirements Traceability in Agent Oriented Development**. Software Engineering for Large-Scale Multi-Agent Systems (SELMAS'02), LNCS, Springer-Verlag Berlin Heidelberg, pp. 57-72, 2003.
- CHESÑEVAR, C.; MCGINNIS, J.; MODGIL, S.; RAHWAN, I.; REED, C.; SIMARI, G.; SOUTH, M.; VREESWIJK, G.; WILLMOTT, S. **Towards an argument interchange format**. The Knowledge Engineering Review, Vol. 21:4, pp. 293–316, 2006.
- CHUNG, L.; NIXON, B.; YU, E.; MYLOPOULOS, J. **Non-Functional Requirements in Software Engineering**. International Series in Software Engineering, Vol. 5, Kluwer, ISBN: 978-0-7923-8666-7, 476 pages, 2000.
- CTS. **Catálogo de Transparência de Software**. Disponível em: [http://www.er.les.inf.puc-rio.br/~wiki/index.php/Transparência\\_de\\_Software](http://www.er.les.inf.puc-rio.br/~wiki/index.php/Transparência_de_Software) (último acesso em Agosto 2011)
- CYNEIROS, L.M. and WERNECK, V.M.B. **An Initial Analysis on How Software Transparency and Trust Influence each other**. 12th Workshop on Requirements Eng., pp. 27-32, 2009.

- CYSNEIRO, G.; ZISMAN A.; SPANOUDAKIS, G. **A Traceability Approach for i\* and UML Models**. 2nd International Workshop on Software Engineering for Large-Scale Multi-Agent Systems (ICSE/SELMAS'03), Oregon, USA, May 2003.
- CYSNEIROS, G. and ZISMAN, A. **Traceability and Completeness Checking for Agent-Oriented Systems**. ACM Symposium on Applied Computing (SAC'08), ISBN: 978-1-59593-753-7, pp. 71-77, Fortaleza, Ceará, Brazil, March 2008.
- CYSNEIROS, L.M.; LEITE, J.C.S.P. **Driving Non-Functional Requirements to Use Cases and Scenarios**. XV Brazilian Symposium on Software Engineering, pp. 7-20, 2001.
- EGYED, A. **A Scenario-Driven Approach to Trace Dependency Analysis**. IEEE Transactions on Software Engineering, ISSN: 0098-5589, Vol. 29, Issue 2, pp. 116-132, February 2003.
- EGYED, A. and GRÜNBAACHER, P. **Automating Requirements Traceability: Beyond the Record & Replay Paradigm**. 17th IEEE International Conference on Automated Software Engineering (ASE'02), ISBN: 0-7695-1736-6, pp. 163-171, September 2002.
- ELAHI, G. and YU, E. **Trust Trade-off Analysis for Security Requirements Engineering**. 17th IEEE International Requirements Engineering Conference (ER'09), pp. 243-248, 2009.
- ELAHI, G.; YU, E.; ZANNONE, N. **A Vulnerability-Centric Requirements Engineering Framework: Analyzing Security Attacks, Countermeasures, and Requirements Based on Vulnerabilities**. Requirements Engineering Journal 15(1), Special Issue - Security Requirements Engineering, pp. 41-62, March 2010.
- FLICKR. **Welcome to Flickr**. Disponível em: <http://www.flickr.com/> (último acesso em Agosto 2011a)
- FLICKR. **ITrace of meeting dd-mm-2011**. Disponível em: <http://www.flickr.com/photos/63908029@N02/5819877406/in/photostream> (último acesso em Junho 2011b)
- FRANCH, X. and MAIDEN, N.A.M. **Modelling Component Dependencies to Inform Their Selection**. COTSBased Software Systems. Springer. LNCS, 2580, pp. 81- 91, 2003.

- FRANCH, X.; GRAU, G.; QUER, C. **A framework for the definition of metrics for actor-dependency models**. IEEE Joint Int. Conf. on Requirements Engineering (RE'04), IEEE Press, pp. 327-328, 2004.
- FREEDOM OF INFORMATION ACT. **United States Department of Justice**. Disponível em: [http://www.usdoj.gov/oip/foia\\_guide07/text\\_foia.pdf](http://www.usdoj.gov/oip/foia_guide07/text_foia.pdf) (último acesso em Setembro 2010)
- GANS, D.; SCHNITZ, D; ARZDORF, T.; JARKE, M.; LAKEMEYER, G. **SNet Reloaded: Roles, Monitoring and Agent Evolution**. International Workshop on Agent-Oriented Information Systems (AOIS'04), Springer, LNCS, 3508, pp. 68-84, 2004.
- GANS, G.; JARKE, M.; LAKEMEYER, G.; SCHMITZ, D. **Deliberation in a Modeling and Simulation Environment for Interorganizational Networks**. 15th International Conference on Advanced Information Systems Engineering (CAiSE'03), LNCS, 2681, pp. 242-257, 2003.
- GIORGINI, P.; MYLOPOULOS, J.; SEBASTIANI, R. **Simple and Minimum-Cost Satisfiability for Goal Models**. 16th International Conference on Advanced Information Systems Engineering (CAiSE'04). Springer, LNCS, 3084, 20-3-5, 2004.
- GIUNCHIGLIA, F.; MYLOPOULOS, J.; PERINI, A. **The Tropos Software Development Methodology: Processes, Models and Diagrams**. AOSE'03, 2003.
- GOGUEN, J. A. **Social Issues in Requirements Engineering**. IEEE International Symposium on Requirements Engineering, pp. 194-195, San Diego, California, USA, January 1993.
- GOTEL, O. and FINKELSTEIN, A. **An Analysis of the Requirements Traceability Problem**. First International Conference on Requirements Engineering, pages 94-101, 1994.
- GOTEL, O. and FINKELSTEIN, A. **Contribution structures**. 2nd Intl. Symp. Requirements Engineering, York, pages 100-107, 1995.
- GROSS, D.; YU, E. **From Non-Functional Requirements to Design through Patterns**. Requirements Engineering Journal Vol. 6, pp. 18-36, 2001.
- GTS. **Grupo de Transparência de Software**. Disponível em: <http://transparencia.les.inf.puc-rio.br/> (último acesso em Janeiro 2011a)

- GTS. **Wiki do Grupo de Transparência de Software**. Disponível em: [http://www.er.les.inf.puc-rio.br/~wiki/index.php/Transparência de Software](http://www.er.les.inf.puc-rio.br/~wiki/index.php/Transparência_de_Software) (último acesso em Agosto 2011b)
- HOLZNER, B. and HOLZNER, L. **Transparency in Global Change: The Vanguard of the Open Society**. University of Pittsburgh Press, 1st edition, Pittsburgh, 2006.
- HORKOFF, J. and YU, E. **Analyzing Goal Models - Different Approaches and How to Choose Among Them**. Requirements Engineering Track - Fourth Edition, part of the 26th ACM Symposium on Applied Computing (SAC'11) 2011.
- HORKOFF, J. and YU, E. **Finding Solutions in Goal Models: An Interactive Backward Reasoning Approach**. 29th International Conference on Conceptual Modeling (ER'10). Springer, LNCS, 6412, pp. 59-75, 2010.
- HORKOFF, J. and YU, E.A. **Qualitative, Interactive Evaluation Procedure for Goal- and Agent-Oriented Models**. 21st International Conference on Advanced Information Systems (CAiSE'09) Forum, CEUR Workshop Proceedings, 2009.
- ISTARWIKI. **Wiki of the Computer Science Group at UofT**. Disponível em: [http://istar.rwth-aachen.de/tiki-view\\_articles.php](http://istar.rwth-aachen.de/tiki-view_articles.php) (último acesso em Maio 2011)
- JURETA, I.; MYLOPOULOS, J.; FAULKNER, S. **Analysis of Multi-Party Agreement in Requirements Validation**. 17th IEEE International Requirements Engineering Conference, IEEE Computer Society, pp. 57-66, 2009.
- KOLP, M.; GIORGINI, P.; MYLOPOULOS, J. **Information Systems Development through Social Structures**. SEKE'02, Ischia, Italy, July 2002.
- KRAMER, B.M.; CHAUDHRI, V.K.; KOUBARAKIS, M.; TOPALOGLOU, T.; WANG, H.; MYLOPOULOS, J. **Implementing Telos**. SIGART Bulletin 2(3): 77-83, 1991.
- LAMSWEERDE, A. van. **Goal-Oriented Requirements Engineering: A Guided Tour**. 5th IEEE International Symposium on RE'01, pp. 249-262, August 2001.

- LATTES. **Plataforma Lattes**. Disponível em: <http://lattes.cnpq.br/> (último acesso em Agosto 2011)
- LATTES-SCHOLAR. **Grupo de Engenharia de Requisitos na PUC-Rio**. Disponível em: <http://www.er.les.inf.puc-rio.br/~wiki/index.php/Lattesscholar> (último acesso em Maio 2011a)
- LATTES-SCHOLAR. **Lattes-Scholar Home Page**. Disponível em: <http://transparencia.les.inf.puc-rio.br:8080/lattesscholarv2/index.lp> (último acesso em Junho 2011b)
- LEHMAN, M. and RAMIL, J. **Software Evolution – Background, Theory, Practice**. Information Processing Letters, Vol. 88, Issues 1-2, pages 33-44, October 2003.
- LEITE, J.C.S. do P. and CAPPELLI, C. **Software Transparency**. Business & Information Systems Engineering: Vol. 2, 2010, Iss. 3, 127-139. Disponível em: <http://aisel.aisnet.org/bise/vol2/iss3/3>
- LEITE, J.C.S. do P.; FREEMAN, P.A. **Requirements validation through viewpoint resolution**. IEEE Transactions Software Engineering 17(12), pp. 1253–1269, 1991.
- LEITE, J.C.S. do P.; ROSSI, G.; BALAGUER, F.; MAIORANA, V.; KAPLAN, G.; HADAD, G.; OLIVEROS, A. **Enhancing a Requirements Baseline with Scenarios**. Requirements Engineering Journal, 2(4):184-198, 1997.
- LEITE, J.C.S. do P.; YU, Y.; LIU, L.; YU, E.S.K.; MYLOPOULOS, J. **Quality-Based Software Reuse**. Advanced Information Systems Engineering Lecture Notes in Computer Science, Volume 3520/2005, pp. 535-550, 2005. DOI: 10.1007/11431855\_37CAiSE 2005: 535-550
- LEITE, J.C.S.P and CAPPELLI, C. **Exploring i\* Characteristics that Support Software Transparency**. 3rd International i\* Workshop, CEUR Workshop Proceedings, Vol. 322, pp. 51-54, 2008. (<http://CEUR-WS.org/Vol-322/>).
- LORD, K. M. **The Perils and Promise of Global Transparency**. State University of New York Press, 2006.
- MAIDEN, N.A.M.; LOCKERBIE, J.; RANDALL, D.; JONES, S.; BUSH, D. **Using Satisfaction Arguments to Enhance i\* Modelling of an Air Traffic Management System**. IEEE International Conference on Requirements Engineering (RE'07), IEEE Press, pp. 49-52, 2007.

- MARCA, D. A.; MCGOWAN, C. L. **SADT: structured analysis and design technique**. McGraw-Hill, ISBN:0-07-040235-3, 392 pages, USA, 1987.
- MCGRATH, M. **Propositions**. In Edward N. Zalta, editor, The Stanford Encyclopedia of Philosophy, Fall, 2008.
- MERCURI, R. **Trusting in transparency**. Communications of the ACM, Vol. 48 Issue 5, pp. 15-19, 2005.
- MEUNIER, P. **Software transparency and purity**. Communications of the ACM, Doi: 10.1145/1314215.1314232, Vol. 51 Issue 2, pp. 104, February 2008.
- MONK, A. and HOWARD, S. **The Rich Picture: A Tool for Reasoning about Work Context**. Methods and Tools, ACM 1072-5220/98/0300, pp. 21-30, April 1998.
- MYLOPOULOS, J. **Goal-Oriented Requirements Engineering**. XI Conferencia Iberoamericana de Software Engineering, pp. 13-17, Brazil, February 2008.
- MYLOPOULOS, J. **DISI – John Mylopoulos**. Disponível em <http://disi.unitn.it/users/john.mylopoulos> (último acesso em Agosto 2011)
- OLIVEIRA, A.P.A.; LEITE, J.C.S. do P.; CYSNEIROS, L.M.. **AGFL - Agent Goals from Lexicon - Eliciting Multi-Agent Systems Intentionality**. 3rd International i\* Workshop, pp. 29-32, 2008.
- ORCHARD, R. **NRC FuzzyJ Toolkit for the Java™ Platform – User’s Guide**. Version 1.10a. Disponível em <http://www.nrc-cnrc.gc.ca/eng/ibp/iit/past-projects/fuzzyj-toolkit.html> (último acesso em Novembro 2009)
- PENSERINI, L.; PERINI, A.; SUSI, A.; MORANDINI, M.; MYLOPOULOS, J. **A Design Framework for Generating BDI Agents from Goal Models**. International Conference on Autonomous Agents, 6th International joint Conference on Autonomous Agents and Multiagent Systems, number 149, 2007. DOI:10.1145/1329125.1329307
- PEREIRA, J. C. R. **Análise de Dados Qualitativos: Estratégias Metodológicas para as Ciências da Saúde, Humanas e Sociais**. ISBN: 85-314-0523-8, Ed. Universidade de São Paulo, 3ª. Edição, 2004.
- PERINI, A. and SUSI, A. **Automating Model Transformations in Agent-Oriented Modeling**. In Agent-Oriented Software Engineering VI, Springer LNCS, vol. 3950, pp. 167-178, Berlin/Heidelberg, 2006.



- PINTO, R.C.; SILVA, C.; CASTRO, J. **A Process for Requirement Traceability in Agent Oriented Development.** Workshop on Requirements Engineering (WER'05), Porto, Portugal, pp. 221-232, 2005.
- PINTO, R.C.; SILVA, C.; CASTRO, J. **Support for Requirement Traceability: The Tropos Case.** 19th Simpósio Brasileiro de Engenharia de Software (SBES'05), Brasil, 2005.
- POHL, K. **PRO-ART: Enabling Requirements Pre-Traceability.** 2nd. IEEE Int. Conference on Requirements Engineering (ICRE'96), ISBN:0-8186-7252-8, pp. 76-84, April 1996.
- POKAHR, A.; BRAUBACH, L.; LAMERSDORF, W. **Jadex: A BDI Reasoning Engine.** Multiagent Systems, Artificial Societies, and Simulated Organizations, vol. 15, pp. 149-174, 2005.
- POKAHR, A and BRAUBACH, L. **Jadex User Guide.** Distributed Systems Group University of Hamburg, Germany, June 2007.
- POLLOCK, J.L. **A Recursive Semantics for Defeasible Reasoning.** Argumentation in Artificial Intelligence, Springer, pp. 173-197, 2009. DOI: 10.1007/978-0-387-98197-0\_9
- PORTER, A.A.; VOTTA Jr., L.G.; BASILI, V.R. **Comparing detection methods for software requirements inspections: A replicated experiment.** IEEE Transactions on Software Engineering, Vol. 21:6, pp. 563-575, 1995.
- PTRANSP. Portal da Transparência do Governo Federal. Disponível em: <http://www.portaltransparencia.gov.br/> (último acesso em Junho 2011)
- RAHWAN, I. and LARSON, K. **Argumentation and Game Theory.** Argumentation in Artificial Intelligence, Springer, pp. 321-339. 2009. DOI: 10.1007/978-0-387-98197-0\_16
- RAMESH, B. and JARKE, M. **Toward Reference Models for Requirements Traceability.** IEEE Transactions on Software Engineering, Vol. 27, No. 1, pages 58-93, January 2001.
- RASHID, A.; SAWYER, P.; MOREIRA, A.; ARAÚJO, J. **Early Aspects: A Model for Aspect-Oriented Requirements Engineering.** 10th Anniversary Joint IEEE International Requirements Engineering Conference (RE'02), pp. 199, 2002.
- SERRANO, M. and LEITE, J.C.S. do P. **A Rich Traceability Model for Social Interactions.** 6th Int. workshop on Traceability in emerging forms of

software engineering. (TEFSE '11). ACM, USA, pp. 63-66, 2011a.

Disponível em: <http://doi.acm.org/10.1145/1987856.1987871>

SERRANO, M.; LEITE, J.C.S.P. **Capturing Transparency-Related Requirements Patterns through Argumentation**, First International Workshop on Requirements Patterns (RePa'11) em conjunto com 19th IEEE International Requirements Engineering Conference (RE'11), Trento, Italy, 2011b.

SERRANO, M.; LEITE, J.C.S.P. **Pré-Rastreabilidade Colaborativa na Engenharia de Requisitos**, Second Workshop of the Brazilian Institute for Web Science Research (Position Paper), Rio de Janeiro, 2011e.

SERRANO, M.; LEITE, J.C.S. do P. **A Social Interaction Based Pre-Traceability for i\* Models**. Fifth International i\* Workshop (iStar'11), Trento, Italy, 2011d.

SERRANO, M.; LEITE, J.C.S. do P. **Development of Agent-Driven Systems: from i\* Architectural Models to Intentional Agents Code**. Fifth International i\* Workshop (iStar'11), Trento, Italy, 2011c.

SERRANO, Maurício; SERRANO, Milene and LEITE, J.C.S. do P. **Dealing with softgoals at runtime**. 2nd International Workshop on Requirements@Run.Time, 2011.

SERRANO, Milene; SERRANO, Maurício; LUCENA, C. J. P. de. **Framework for Content Adaptation in Ubiquitous Computing centered on Agents Intentionality and Collaborative MAS**. 4th Workshop on Software Engineering for Agent-Oriented Systems (SEAS'08), SBES'08, pp. 1-12, Campinas, Brasil, Outubro 2008c.

SERRANO, Milene; SERRANO, Maurício; LUCENA, C. J. P. de. **Ubiquitous Software Development Driven by Agents' Intentionality**. 11<sup>th</sup> International Conference on Enterprise Information Systems (ICEIS'09), vol. SAIC, pp. 25-34, Milan, Italy, May 2009.

SERRANO, Milene; SERRANO, Maurício; NAPOLITANO, F.; KINDER, E.; DOUGLAS, M.; LOYOLA, D.; REZENDE, B.; LEITE, J.C.S. do P. **A Proposal for the Evaluation of Requirements Teams**. 11th Workshop on Requirements Engineering (WER'08), pp. 34-44, Barcelona, Spain, 2008. In Portuguese: Uma Proposta para Avaliação de Equipes de Requisitos.

- Décimo Primeiro Workshop em Engenharia de Requisitos (WER'08), pp. 34-44, Barcelona, Espanha, 2008a.
- SERRANO, Milene; SERRANO, Maurício; NAPOLITANO, F.; KINDER, E.; DOUGLAS, M.; LOYOLA, D.; REZENDE, B.; LEITE, J.C.S. do P. **Avaliação Experimental de um Método para Avaliação de Equipes de Requisitos**. Simpósio Brasileiro de Engenharia de Software (SBES'08), Campinas, SP, Brasil, 2008b.
- SILVA, C.; DIAS, P.; ARAÚJO J.; MOREIRA, A. **De Architecturas Organizacionais em i\* a Architecturas Baseadas em Agentes: Uma abordagem orientada a modelos**. XIV Workshop on Requirements Engineering, page 48, April 2011.
- SOX. **Sarbanes-Oxley Act of 2002**. Pub. L. No. 107-204, 116 Stat. 745 (codified as amended in scattered sections of 15 U.S.C.), 2002.
- SUPAKKUL, S.; HILL, T.; CHUNG, L.; TUN, T.T.; LEITE, J.C.S. do P. **An NFR Pattern Approach to Dealing with NFRs**. 18th IEEE International Requirements Engineering Conference (RE'10), pp. 179-188, ISBN: 978-0-7695-4162-4, Sydney, Australia, September-October 2010.
- TB. **Transparência Brasil**. 2000. Disponível em: <http://www.transparencia.org.br/index.html> (último acesso em Setembro 2010)
- TI. **Transparency International**. 1993. Disponível em: <http://www.transparency.org/> (último acesso em Setembro 2010)
- TORANZO, M.A. **Um Framework para Melhorar o Rastreamento de Requisitos**. Dissertação de Doutorado, Centro de Informática da Universidade Federal de Pernambuco-UFPE, Brasil, 2002.
- TRAVASSOS, G. H.; GUROV, D.; AMARAL, E. A. G. do. **Introdução à Engenharia de Software Experimental**. Relatório Técnico RT-ES-590/02, Programa de Engenharia de Sistemas e Computação, COPPE/UFRJ, 52 pgs, Janeiro 2001.
- VAN LAMSWEERDE, A. **Goal-Oriented Requirements Engineering: A Guided Tour**. 5th IEEE International Symposium on Requirements Engineering (RE'01), Publisher IEEE Computer Society, ISBN: 0-7695-1125-2, Washington, DC, USA, p.249, August 2001.

- VAN LAMSWEERDE, A. **Reasoning about alternative requirements options.** Conceptual Modeling: Foundations and Applications, A. T. Borgida, V. K. Chaudhri, P. Giorgini, and E. S. Yu, Eds., pp. 380-397, 2009.
- UML. **Object Management Group – UML.** Disponível em <http://www.uml.org/> (último acesso em Agosto 2011)
- WERPAPERS. **WERpapers - Artigos Publicados no Workshop em Engenharia de Requisitos.** Disponível em <http://wer.inf.puc-rio.br/WERpapers/> (ultimo acesso em Agosto 2011)
- WOOLDRIDGE, M. **An Introduction to MultiAgent Systems.** John Wiley & Sons Ltd, ISBN 0-471-49691-X, 366 pages, 2002.
- XIE, H.; LIU, L.; YANG, J. **i\*-prefer: optimizing requirements elicitation process based on actor preferences.** ACM Symposium on Applied Computing (SAC'09), pp. 347-354, New York, NY, USA, 2009.
- YU, E. **Modelling strategic relationships for process reengineering.** PhD Thesis, ISBN:0-612-02887-9, Order Number:AAINN02887, 181 pages, University of Toronto Toronto, Canada, 1995.
- YU, E.S.K. **Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering.** 3rd IEEE International Symposium on Requirements Engineering (RE'97), IEEE Computer Society, USA, pp. 226-235, 1997.
- ZADEH, L.A. **Fuzzy Sets - Information and Control.** vol.8, pp. 338-352, 1965.

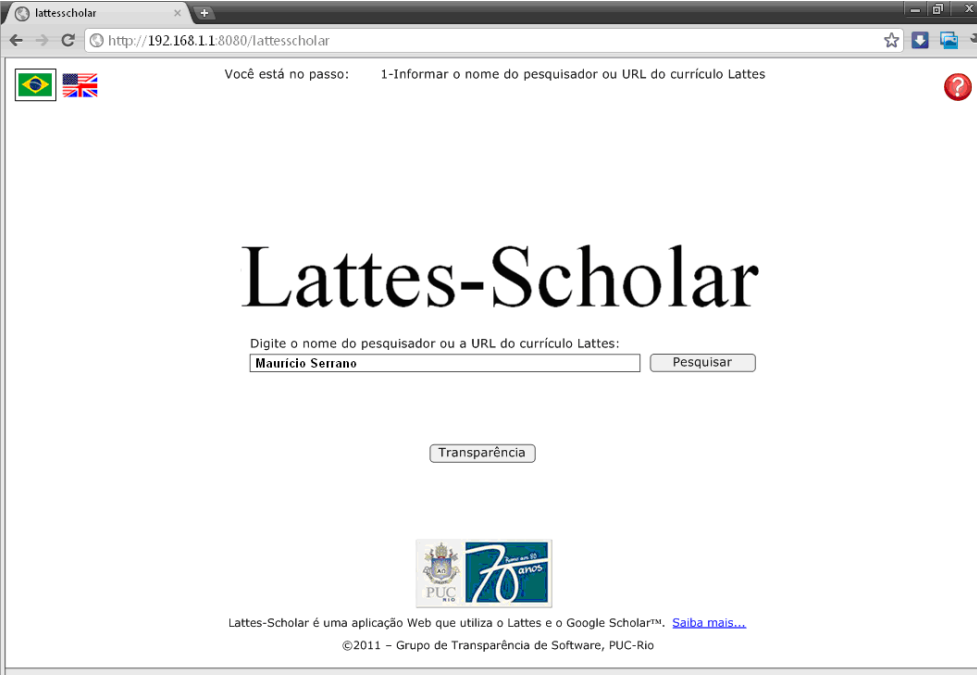
## Apêndice A. Exemplos de Uso do Lattes-Scholar

*Nesse apêndice apresentamos dois exemplos de uso do Lattes-Scholar. O primeiro exemplo mostra as telas do Lattes-Scholar quando o consumidor entra com o nome de um pesquisador. O segundo exemplo mostra as telas do Lattes-Scholar quando o consumidor entra com a URL do currículo lattes de um pesquisador.*

O consumidor, ao utilizar o Lattes-Scholar, pode iniciar o processo de duas formas: fornecendo, na página principal, o nome do pesquisador ou a URL do currículo do pesquisador.

### A.1 Processo Iniciado pelo Consumidor com o Nome do Pesquisador

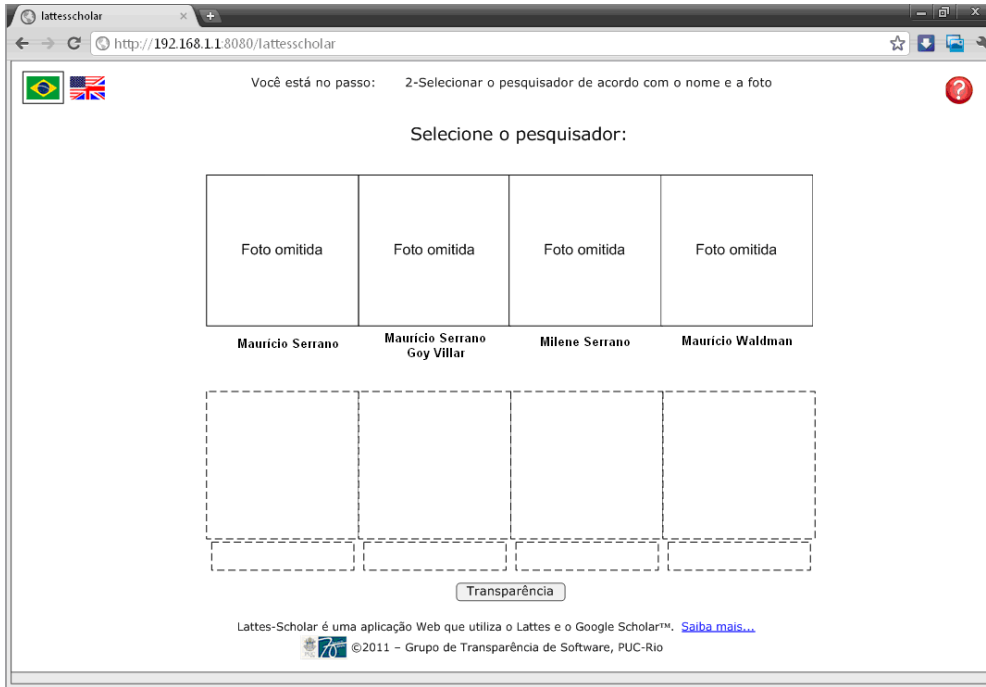
O consumidor deve digitar o nome do pesquisador na caixa de texto e clicar em “Pesquisar”. A tela abaixo, o passo um, descreve esse processo.



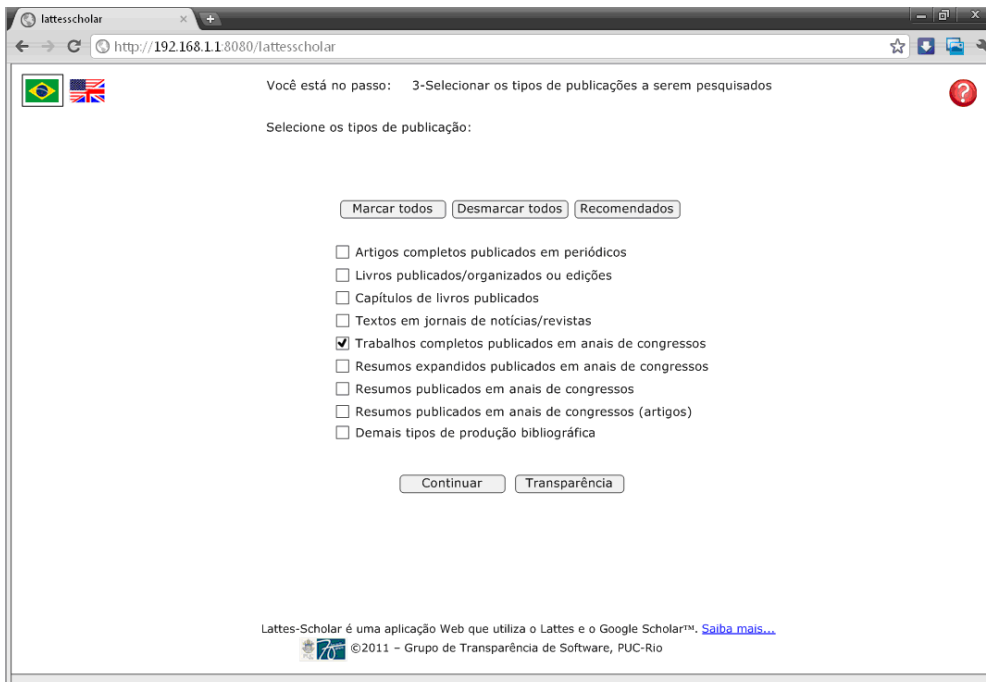
The screenshot shows a web browser window with the URL `http://192.168.1.1:8080/lattesscholar`. The page content includes:

- Flags for Brazil and the United Kingdom.
- Progress indicator: "Você está no passo: 1-Informar o nome do pesquisador ou URL do currículo Lattes".
- Large heading: "Lattes-Scholar".
- Input field: "Digite o nome do pesquisador ou a URL do currículo Lattes:" with the text "Maurício Serrano" entered.
- Search button: "Pesquisar".
- Transparency button: "Transparência".
- Logos for PUC-Rio and its 70th anniversary.
- Footer text: "Lattes-Scholar é uma aplicação Web que utiliza o Lattes e o Google Scholar™. Saiba mais... ©2011 - Grupo de Transparência de Software, PUC-Rio".

O Lattes-Scholar recupera do Lattes os nomes e fotos dos pesquisadores que possuem o nome semelhante ao nome digitado pelo Consumidor. A tela abaixo, o passo dois, apresenta pesquisadores com nome semelhante ao nome informado, “Maurício Serrano”. O Consumidor seleciona o pesquisador pela foto.



As seções do currículo do pesquisador selecionado são recuperadas e apresentadas ao Consumidor. Na tela abaixo, o passo três, o Consumidor deve selecionar os tipos de publicação a serem pesquisadas.



O Lattes-Scholar agrupa todas as publicações contidas nas seções selecionadas e ordena o grupo pelo número de citações. Uma vez ordenadas pelo número de citações em ordem decrescente, as publicações são apresentadas ao Consumidor em uma tabela. A tela abaixo, o passo quatro, apresenta a tabela de publicações ordenada e o h-index para essa seleção.

Você está no passo: 4-Visualizar as publicações ordenadas pelo número de citações

Resultado:

**H-index dessa lista: 1**

Publicações de <b>Maurício Serrano</b>	Citações
SERRANO, Milene; SERRANO, Maurício; LUCENA, C. J. P. de Framework for Content Adaptation in Ubiquitous Computing centered on Agents Intentionality and Collaborative MAS. 4th Workshop on Software Engineering for Agent-Oriented Systems (SEAS'08), SBES'08, pp. 1-12, Campinas, Brasil, Outubro 2008.	1
SERRANO, Milene; SERRANO, Maurício; NAPOLITANO, F.; KINDER, E.; DOUGLAS, M.; LOYOLA, D.; REZENDE, B.; LEITE, J.C.S. do P. Avaliação Experimental de um Método para Avaliação de Equipes de Requisitos. Simpósio Brasileiro de Engenharia de Software (SBES'08), Campinas, SP, Brasil, 2008.	1
SERRANO, M.; LEITE, J.C.S.P. Capturing Transparency-Related Requirements Patterns through Argumentation, First International Workshop on Requirements Patterns (RePa'11) em conjunto com 19th IEEE International Requirements Engineering Conference (RE'11), Trento, Italy, 2011.	0
SERRANO, Maurício; SERRANO, Milene and LEITE, J.C.S. do P. Dealing with softgoals at runtime. 2nd International Workshop on Requirements@Run.Time, 2011.	0
SERRANO, M. and LEITE, J.C.S. do P. A Rich Traceability Model for Social Interactions. 8th Int. workshop on Traceability in emerging forms of software engineering. (TEFSE'11). ACM, USA, pp. 63-66, 2011.	0
SERRANO, M.; LEITE, J.C.S. do P. Development of Agent-Driven Systems: from P Architectural Models to Intentional Agents Code. Fifth International P Workshop (Star'11), Trento, Italy, 2011.	0
SERRANO, Milene; SERRANO, Maurício; NAPOLITANO, F.; KINDER, E.; DOUGLAS, M.; LOYOLA, D.; REZENDE, B.; LEITE, J.C.S. do P. A Proposal for the Evaluation of Requirements Teams. 11th Workshop on Requirements Engineering (WER'08), pp. 34-44, Barcelona, Spain, 2008.	0
SERRANO, M.; LEITE, J.C.S. do P. A Social Interaction Based Pre-Traceability for P Models. Fifth International P Workshop (Star'11), Trento, Italy, 2011.	0
SERRANO, M.; LEITE, J.C.S. P. <i>Tratado de Rastreabilidade Colaborativa na Engenharia de Requisitos</i> . Second	

Voltar Nova consulta Transparência

Lattes-Scholar é uma aplicação Web que utiliza o Lattes e o Google Scholar™. [Saiba mais...](#)

©2011 - Grupo de Transparência de Software, PUC-Rio

A tela acima conclui o processo. É possível voltar (botão “Voltar”) à tela anterior para selecionar outros tipos de publicação ou realizar uma nova consulta (botão “Nova Consulta”), retornando à página principal. Vale ressaltar que em todas as telas, o Consumidor pode trocar de idioma, português ou inglês, clicando na respectiva bandeira. O Consumidor também pode solicitar ajuda clicando no ícone de interrogação, obtendo instruções para a realização do processo.

Os botões “Transparência”, presentes em todas as telas, permitem ao Consumidor visualizar o rastro/história do processo. Esse rastro/história é composto pelo passo atual, os passos já realizados e as informações fornecidas.

## A.2

### Processo Iniciado pelo Consumidor com a URL do Currículo Lattes do Pesquisador

O consumidor deve colar a URL do currículo do pesquisador na caixa de texto e clicar em “Pesquisar”. A tela a seguir, o passo um, descreve esse processo.

lattes-scholar


http://192.168.1.1:8080/lattes-scholar

Você está no passo: 1-Informar o nome do pesquisador ou URL do currículo Lattes

Brasil Reino Unido

# Lattes-Scholar

Digite o nome do pesquisador ou a URL do currículo Lattes:



Lattes-Scholar é uma aplicação Web que utiliza o Lattes e o Google Scholar™. [Saiba mais...](#)

©2011 - Grupo de Transparência de Software, PUC-Rio

As seções do currículo do pesquisador informado são recuperadas e apresentadas ao Consumidor. Não é necessária uma tela para a seleção do pesquisador. Na tela abaixo, o passo dois, o Consumidor deve selecionar os tipos de publicação a serem pesquisadas.

lattes-scholar

http://192.168.1.1:8080/lattes-scholar


Você está no passo: 2-Selecionar os tipos de publicações a serem pesquisados

Brasil Reino Unido

Selecione os tipos de publicação:

- Artigos completos publicados em periódicos
- Livros publicados/organizados ou edições
- Capítulos de livros publicados
- Textos em jornais de notícias/revistas
- Trabalhos completos publicados em anais de congressos
- Resumos expandidos publicados em anais de congressos
- Resumos publicados em anais de congressos
- Resumos publicados em anais de congressos (artigos)
- Demais tipos de produção bibliográfica

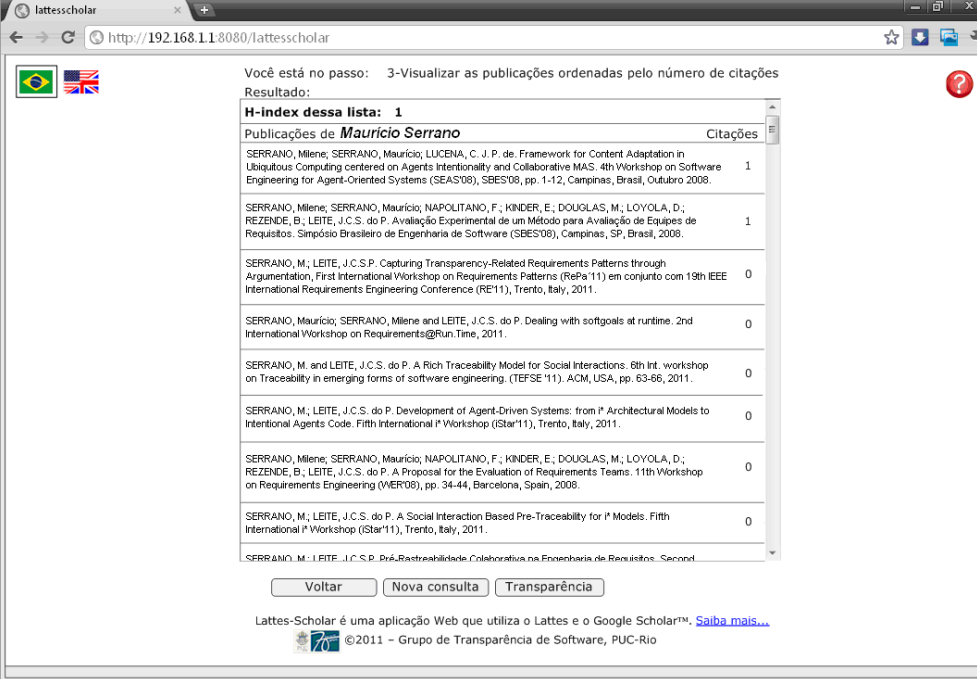
Lattes-Scholar é uma aplicação Web que utiliza o Lattes e o Google Scholar™. [Saiba mais...](#)

 ©2011 - Grupo de Transparência de Software, PUC-Rio

O Lattes-Scholar agrupa todas as publicações contidas nas seções selecionadas e ordena o grupo pelo número de citações. Uma vez ordenadas pelo número de citações em ordem decrescente, as publicações são apresentadas ao



Consumidor em uma tabela. A tela abaixo, o passo três, apresenta a tabela de publicações ordenada e o h-index para essa seleção.



Você está no passo: 3-Visualizar as publicações ordenadas pelo número de citações

Resultado:

**H-index dessa lista: 1**

Publicações de <b>Maurício Serrano</b>	Citações
SERRANO, Milene; SERRANO, Maurício; LUCENA, C. J. P. de. Framework for Content Adaptation in Ubiquitous Computing centered on Agents Intentionality and Collaborative MAS. 4th Workshop on Software Engineering for Agent-Oriented Systems (SEAS'08), SBES'08, pp. 1-12, Campinas, Brasil, Outubro 2008.	1
SERRANO, Milene; SERRANO, Maurício; NAPOLITANO, F.; KINDER, E.; DOUGLAS, M.; LOYOLA, D.; REZENDE, B.; LEITE, J.C.S. do P. Avaliação Experimental de um Método para Avaliação de Equipes de Requisitos. Simpósio Brasileiro de Engenharia de Software (SBES'08), Campinas, SP, Brasil, 2008.	1
SERRANO, M.; LEITE, J.C.S.P. Capturing Transparency-Related Requirements Patterns Through Argumentation, First International Workshop on Requirements Patterns (RePa'11) em conjunto com 19th IEEE International Requirements Engineering Conference (RE'11), Trento, Italy, 2011.	0
SERRANO, Maurício; SERRANO, Milene and LEITE, J.C.S. do P. Dealing with softgoals at runtime. 2nd International Workshop on Requirements@Run Time, 2011.	0
SERRANO, M. and LEITE, J.C.S. do P. A Rich Traceability Model for Social Interactions. 6th Int. workshop on Traceability in emerging forms of software engineering. (TEFSE'11). ACM, USA, pp. 63-66, 2011.	0
SERRANO, M.; LEITE, J.C.S. do P. Development of Agent-Driven Systems: from P Architectural Models to Intentional Agents Code. Fifth International P Workshop (ISar'11), Trento, Italy, 2011.	0
SERRANO, Milene; SERRANO, Maurício; NAPOLITANO, F.; KINDER, E.; DOUGLAS, M.; LOYOLA, D.; REZENDE, B.; LEITE, J.C.S. do P. A Proposal for the Evaluation of Requirements Teams. 11th Workshop on Requirements Engineering (WER'08), pp. 34-44, Barcelona, Spain, 2008.	0
SERRANO, M.; LEITE, J.C.S. do P. A Social Interaction Based Pre-Traceability for P Models. Fifth International P Workshop (ISar'11), Trento, Italy, 2011.	0
SERRANO, M.; LEITE, J.C.S.P. Pré-Rastreabilidade Colaborativa na Engenharia de Requisitos. Second	

Voltar Nova consulta Transparência

Lattes-Scholar é uma aplicação Web que utiliza o Lattes e o Google Scholar™. [Saiba mais...](#)

©2011 - Grupo de Transparência de Software, PUC-Rio

## Apêndice B. Código-Fonte da Máquina de Raciocínio Nebulosa para Metas Flexíveis

*Nesse apêndice apresentamos o código-fonte da máquina de raciocínio nebulosa que capacita agentes intencionais a raciocinarem em termos de metas flexíveis. Também são apresentados os códigos-fonte das classes necessárias para representar metas flexíveis, tarefas e contribuições.*

### B.1 Código-fonte da Classe MetaPlan

Esta seção apresenta o código-fonte da classe MetaPlan, pertencente ao pacote “*istar.metamodel*”. Essa classe estende a classe Plan do *framework* JADEX e implementa a máquina de raciocínio nebulosa para metas flexíveis através do método `runReasoningEngine(int runMode)`. Esse método é chamado no método `body()`, onde o parâmetro `runMode` é passado com o valor zero, desabilitando os efeitos colaterais, ou o valor um, habilitando-os. Por efeitos colaterais entendemos os impactos nas metas flexíveis ao **não** se executar uma determinada tarefa.

A classe MetaPlan deve ser utilizada em todo momento que o agente precisa escolher uma entre várias tarefas para se atingir uma meta, ou seja, toda vez que é disparado um *metagoal* pela máquina de raciocínio BDI do Jadex.

A importação de biblioteca “`import nrc.fuzzy.*;`” faz uma referência à API JAVA disponibilizada no National Research Council, do Canadá.

```
package istar.metamodel;
```

```
import jadex.adapter.fipa.AgentDescription;
import jadex.adapter.fipa.AgentIdentifier;
import jadex.adapter.fipa.SFipa;
import jadex.adapter.fipa.SearchConstraints;
import jadex.adapter.fipa.ServiceDescription;
import jadex.runtime.ICandidateInfo;
import jadex.runtime.IExpression;
import jadex.runtime.IGoal;
```

```

import jadex.runtime.IMessageEvent;
import jadex.runtime.Plan;
import java.util.Enumeration;
import java.util.Vector;
import nrc.fuzzy.*;

/**
 * @author Maurício Serrano
 */

public class MetaPlan extends Plan {

    public MetaPlan() {

    }

    public void runReasoningEngine(int runMode) {
        ICandidateInfo[] alternatives = (ICandidateInfo[])
            getParameterSet("applicables").getValues();
        IExpression queryTask = getExpression("associate_task");
        IExpression querySoftgoal = getExpression("associate_softgoal");

        assert alternatives.length > 0;
        System.out.println("Number of alternatives:" + alternatives.length);

        ICandidateInfo selected = null;
        Task[] tasks = new Task[alternatives.length];
        Vector allRules = new Vector(16);
        Vector inputs = new Vector(2);

        Softgoal[] allSoftgoals = (Softgoal[])
            this.getBeliefbase().getBeliefSet("softgoals").getFacts();
        Vector impactedSoftgoalsLabels = new Vector(1);
        Vector impactedSoftgoals = new Vector(1);

        int dummyTaskIndex = -1;

        // turns on or off the colateral effects of not selecting a task
        if (runMode == 1) {
            System.out.println("RunMode: Considering colateral effects.");
        } else {
            System.out.println("RunMode: Colateral effects being ignored.");
        }

        // getting to know the alternatives, the impacted softgoals
        // and generating the fuzzy rules according to the current context
        for (int i = 0; i < alternatives.length; i++) {
            String planName = alternatives[i].getPlan().getName();
            planName = planName.substring(0, planName.length() - 2);
            Task tempTask = (Task) queryTask.execute("$planname", planName);
            tasks[i] = tempTask;
        }
    }
}

```

```

try {
    if (tempTask == null) {
        if (planName.contains("Dummy")) {
            System.out.println("Dummy task found.");
            dummyTaskIndex = i;
        } else {
            System.out.println("Invalid task name.");
        }
    } else {
        for (int j = 0; j < tempTask.getSoftgoalContributions().length; j++) {

            // getting the two fuzzy variables and the contribution
            // involved in each impact
            FuzzyVariable contributor = tempTask.getFuzzyVariable();
            ContributionType contributionType =
tempTask.getSoftgoalContributions()[j].getContributionType();
            String softgoalName =
tempTask.getSoftgoalContributions()[j].getContributed();
            Object softgoal = querySoftgoal.execute("$softgoalname",
softgoalName);
            if (softgoal instanceof DecomposedSoftgoal) {
                this.fail();
            }
            LeafSoftgoal tempSoftgoal = (LeafSoftgoal) softgoal;
            FuzzyVariable contributed = tempSoftgoal.getFuzzyVariable();

            // discovering all the impacted softgoals in this means-end
            if (!impactedSoftgoalsLabels.contains(softgoalName)) {
                impactedSoftgoalsLabels.addElement(softgoalName);
                tempSoftgoal.setAlternativeImpactsSize(alternatives.length);
                impactedSoftgoals.addElement(tempSoftgoal);
                System.out.println("New impacted softgoal: " + softgoalName);
            } else {
                System.out.println("Softgoal already in the list.");
            }

            // creating the 4 rules according to contribution type
            // for each impact
            FuzzyRule tempRule1 = new FuzzyRule();
            FuzzyRule tempRule2 = new FuzzyRule();
            FuzzyRule tempRule3 = new FuzzyRule();
            FuzzyRule tempRule4 = new FuzzyRule();
            FuzzyRule tempRule5 = new FuzzyRule();
            tempRule5.addAntecedent(new FuzzyValue(contributor,
"undecided"));
            tempRule5.addConclusion(new FuzzyValue(contributed,
"undecided"));

            switch (contributionType) {

```

```

        case MAKE:
            tempRule1.addAntecedent(new FuzzyValue(contributor,
"satisfied"));
            tempRule1.addConclusion(new FuzzyValue(contributed,
"satisfied"));
            tempRule2.addAntecedent(new FuzzyValue(contributor,
"partially_satisfied"));
            tempRule2.addConclusion(new FuzzyValue(contributed,
"partially_satisfied"));
            tempRule3.addAntecedent(new FuzzyValue(contributor,
"partially_denied"));
            tempRule3.addConclusion(new FuzzyValue(contributed,
"partially_denied"));
            tempRule4.addAntecedent(new FuzzyValue(contributor,
"denied"));
            tempRule4.addConclusion(new FuzzyValue(contributed,
"denied"));
            break;
        case HELP:
            tempRule1.addAntecedent(new FuzzyValue(contributor,
"satisfied"));
            tempRule1.addConclusion(new FuzzyValue(contributed,
"partially_satisfied"));
            tempRule2.addAntecedent(new FuzzyValue(contributor,
"partially_satisfied"));
            tempRule2.addConclusion(new FuzzyValue(contributed,
"partially_satisfied"));
            tempRule3.addAntecedent(new FuzzyValue(contributor,
"partially_denied"));
            tempRule3.addConclusion(new FuzzyValue(contributed,
"partially_denied"));
            tempRule4.addAntecedent(new FuzzyValue(contributor,
"denied"));
            tempRule4.addConclusion(new FuzzyValue(contributed,
"partially_denied"));
            break;
        case HURT:
            tempRule1.addAntecedent(new FuzzyValue(contributor,
"satisfied"));
            tempRule1.addConclusion(new FuzzyValue(contributed,
"partially_denied"));
            tempRule2.addAntecedent(new FuzzyValue(contributor,
"partially_satisfied"));
            tempRule2.addConclusion(new FuzzyValue(contributed,
"partially_denied"));
            tempRule3.addAntecedent(new FuzzyValue(contributor,
"partially_denied"));
            tempRule3.addConclusion(new FuzzyValue(contributed,
"partially_satisfied"));

```

```

        tempRule4.addAntecedent(new FuzzyValue(contributor,
"denied"));
        tempRule4.addConclusion(new FuzzyValue(contributed,
"partially_satisfied"));
        break;
    case BREAK:
        tempRule1.addAntecedent(new FuzzyValue(contributor,
"satisfied"));
        tempRule1.addConclusion(new FuzzyValue(contributed,
"denied"));
        tempRule2.addAntecedent(new FuzzyValue(contributor,
"partially_satisfied"));
        tempRule2.addConclusion(new FuzzyValue(contributed,
"partially_denied"));
        tempRule3.addAntecedent(new FuzzyValue(contributor,
"partially_denied"));
        tempRule3.addConclusion(new FuzzyValue(contributed,
"partially_satisfied"));
        tempRule4.addAntecedent(new FuzzyValue(contributor,
"denied"));
        tempRule4.addConclusion(new FuzzyValue(contributed,
"partially_satisfied"));
        break;
    default:
        System.out.println("Not supported yet");
        break;
    }
    allRules.addElement(tempRule1);
    allRules.addElement(tempRule2);
    allRules.addElement(tempRule3);
    allRules.addElement(tempRule4);
    allRules.addElement(tempRule5);
}
}
} catch (Exception ex) {
    ex.printStackTrace();
}
}

```

```
System.out.println("Number of rules:" + allRules.size());
```

```
// preparing to receive the result for each alternative
double[] results = new double[alternatives.length];
```

```
// considering each alternative
```

```
for (int i = 0; i < alternatives.length; i++) {
    System.out.println("\nAlternativa: " + (i + 1));
```

```
    if (i == dummyTaskIndex) {
        System.out.println("\nAlternativa Dummy.");
```

```

} else {

    // marking the task as satisfied and the others as denied - mode 1
    // marking the task as satisfied and the others as neutral - mode 0
    inputs.removeAllElements();
    for (int j = 0; j < alternatives.length; j++) {
        try {
            if (j != dummyTaskIndex) {
                if (i == j) {
                    FuzzyValue.setConfineFuzzySetsToUOD(true);
                    inputs.addElement(new
FuzzyValue(tasks[j].getFuzzyVariable(), new TriangleFuzzySet(100.0, 5.0)));
                    FuzzyValue.setConfineFuzzySetsToUOD(false);
                    //System.out.println("Added Input:");
                    //System.out.println(((FuzzyValue)
inputs.lastElement()).plotFuzzyValue("+"));
                } else {
                    if (runMode == 0) {
                        FuzzyValue.setConfineFuzzySetsToUOD(true);
                        inputs.addElement(new
FuzzyValue(tasks[j].getFuzzyVariable(), new TriangleFuzzySet(0.0, 1.0)));
                        FuzzyValue.setConfineFuzzySetsToUOD(false);
                        //System.out.println("Added Input:");
                        //System.out.println(((FuzzyValue)
inputs.lastElement()).plotFuzzyValue("+"));
                    } else {
                        FuzzyValue.setConfineFuzzySetsToUOD(true);
                        inputs.addElement(new
FuzzyValue(tasks[j].getFuzzyVariable(), new TriangleFuzzySet(-100.0, 5.0)));
                        FuzzyValue.setConfineFuzzySetsToUOD(false);
                        //System.out.println("Added Input:");
                        //System.out.println(((FuzzyValue)
inputs.lastElement()).plotFuzzyValue("+"));
                    }
                }
            }
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }

    // set the correct task as input for each rule
    for (Enumeration e1 = allRules.elements(); e1.hasMoreElements();) {
        FuzzyRule tempFuzzyRule = (FuzzyRule) e1.nextElement();
        tempFuzzyRule.removeAllInputs();

        String inputName =
tempFuzzyRule.antecedentAt(0).getFuzzyVariable().getName();
        for (int j = 0; j < inputs.size(); j++) {

```

```

        String taskName = ((FuzzyValue)
inputs.elementAt(j)).getFuzzyVariable().getName();
        if (inputName.equalsIgnoreCase(taskName)) {
            tempFuzzyRule.addInput((FuzzyValue) inputs.elementAt(j));
            break;
        }
    }
}

// run
FuzzyValueVector rulesImpacts = new FuzzyValueVector();
for (Enumeration e = allRules.elements(); e.hasMoreElements();) {
    FuzzyRule tempFuzzyRule = (FuzzyRule) e.nextElement();

    try {
        if (tempFuzzyRule.testRuleMatching()) {
            if
(tempFuzzyRule.antecedentAt(0).getLinguisticExpression().equalsIgnoreCase(
"undecided")) {
                System.out.println("Impact discarded.");
            } else {
                System.out.println("Rule: if " +
tempFuzzyRule.antecedentAt(0).getFuzzyVariable().getName() + " is " +
tempFuzzyRule.antecedentAt(0).getLinguisticExpression() + " then " +
tempFuzzyRule.conclusionAt(0).getFuzzyVariable().getName() + " is " +
tempFuzzyRule.conclusionAt(0).getLinguisticExpression() + ".");
                FuzzyValueVector fvv = tempFuzzyRule.execute();

                fvv.fuzzyValueAt(0).setLinguisticExpression(tempFuzzyRule.conclusionAt(0).
getLinguisticExpression());

                System.out.println(fvv.fuzzyValueAt(0).plotFuzzyValue("+"));
                rulesImpacts.concat(fvv);
            }
        } else {
            System.out.println("Not match.");
        }

    } catch (Exception ex) {
        ex.printStackTrace();
    }
}

// assign each impact to its corresponding softgoal
// the softgoal order in the result is the same in the
// impactedSoftgoalsLabels vector
for (int j = 0; j < rulesImpacts.size(); j++) {
    String impactedSoftgoalName =
rulesImpacts.fuzzyValueAt(j).getFuzzyVariable().getName();
}

```



```

        for (int k = 0; k < impactedSoftgoalsLabels.size(); k++) {
            String storedSoftgoalName = (String)
impactedSoftgoalsLabels.elementAt(k);
            if
(impactedSoftgoalName.equalsIgnoreCase(storedSoftgoalName)) {
                LeafSoftgoal softgoal = (LeafSoftgoal)
impactedSoftgoals.elementAt(k);
                softgoal.addImpactToAlternative(rulesImpacts.fuzzyValueAt(j),
i);
                    break;
                }
            }
        }

        // print the result
        try {
            LeafSoftgoal softgoal1 = (LeafSoftgoal)
impactedSoftgoals.elementAt(0);
            System.out.println("Number of Impacts: " +
softgoal1.getAlternativeImpacts(i).size());
            System.out.println("Moment: " +
softgoal1.getAlternativeImpacts(i).momentDefuzzify());
            //System.out.println("Center Of Area: " +
fvv.centerOfAreaDefuzzify());
            //System.out.println("Maximum: " + fvv.maximumDefuzzify());
            //System.out.println("Weighted Average: " +
fvv.weightedAverageDefuzzify());
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }

    // analyze
    double highestValue = Double.MIN_VALUE;
    int selectedAlternative = -1;
    selected = null;
    System.out.println("\nNumber of Results: " + results.length);
    for (int i = 0; i < alternatives.length; i++) {
        double value = 0.0;

        if (i != dummyTaskIndex) {
            System.out.println("Results for " + tasks[i].getTaskName() + ":");
            for (int k = 0; k < allSoftgoals.length; k++) {
                Softgoal softgoal = allSoftgoals[k];
                try {
                    value += softgoal.getWeight() *
softgoal.getValueForAlternative(i);
                } catch (Exception ex) {
                    ex.printStackTrace();
                }
            }
        }
    }

```

```

    }
  }
  System.out.println("Value: " + value);
  results[i] = value;
  if (value > highestValue) {
    highestValue = value;
    selectedAlternative = i;
    selected = alternatives[i];
  }
}
}

// find all agents
AgentIdentifier[] filteredReceivers = null;
IGoal df_search = createGoal("df_search");
AgentDescription desc = new AgentDescription();
desc.addService(new ServiceDescription("softgoal_control", "softgoal",
null));
df_search.getParameter("description").setValue(desc);
SearchConstraints sc = new SearchConstraints();
sc.setMaxResults(100);
df_search.getParameter("constraints").setValue(sc);
dispatchSubgoalAndWait(df_search);

AgentDescription[] result = (AgentDescription[])
df_search.getParameterSet("result").getValues();
int myNameIndex = -1;

if (result.length > 1) {

  AgentIdentifier[] receivers = new AgentIdentifier[result.length];
  for (int j = 0; j < result.length; j++) {
    receivers[j] = result[j].getName();
    String completeName = receivers[j].getLocalName() + "@" +
receivers[j].getPlatformName();
    if
(completeName.equalsIgnoreCase(this.getAgentIdentifier().getName())) {
      myNameIndex = j;
    }
  }

  // removing this agent from the receivers
  filteredReceivers = new AgentIdentifier[result.length - 1];
  for (int k=0; k<result.length-1; k++) {
    filteredReceivers[k] = receivers[k];
  }
  if (myNameIndex < (result.length-1))
    filteredReceivers[myNameIndex] = receivers[result.length-1];
} else {
  System.out.println("No other softgoals controlers found in DF.");
}

```

```

    }

    // apply the impacts
    for (int i = 0; i < impactedSoftgoals.size(); i++) {
        LeafSoftgoal softgoal = (LeafSoftgoal) impactedSoftgoals.elementAt(i);
        FuzzyValueVector alternativeNewImpacts =
        softgoal.getAlternativeNewImpacts(selectedAlternative);

        String string = "Softgoal " + softgoal.getSoftgoalName();
        string = string + " received " + alternativeNewImpacts.size() + "
        impact(s).";
        System.out.println(string);

        if (filteredReceivers != null) {
            String[] impacts = new String[alternativeNewImpacts.size()];
            for (int j=0; j<alternativeNewImpacts.size(); j++) {
                impacts[j] =
                alternativeNewImpacts.fuzzyValueAt(j).getLinguisticExpression();

                //System.out.println(alternativeNewImpacts.fuzzyValueAt(j).plotFuzzyValue("
                +"));
            }

            if (alternativeNewImpacts.size() != 0) {
                SoftgoalImpacts softgoalImpacts = new
                SoftgoalImpacts(softgoal.getSoftgoalName(), impacts);

                // broadcast message to all agents
                System.out.println("Sending message(s) to " +
                filteredReceivers.length + " agent(s).");
                IMessageEvent message =
                createMessageEvent("inform_impacted_softgoal");

                message.getParameterSet(SFipa.RECEIVERS).addValues(filteredReceivers);

                message.getParameter(SFipa.CONTENT).setValue(softgoalImpacts);

                this.sendMessage(message);
            }

            softgoal.applyAlternative(selectedAlternative);
        }

        this.getParameterSet("result").addValue(selected);
    }

    @Override
    public void passed() {

```

```

        //System.out.println("Passed");
    }

    @Override
    public void failed() {
        // Clean-up code for plan failure.
        System.out.println("Failed");
        getException().printStackTrace();
    }

    @Override
    public void aborted() {
        // Clean-up code for an aborted plan.
        System.out.println("Aborted");
        System.out.println("Goal achieved? "+isAbortedOnSuccess());
    }

    @Override
    public void body() {
        System.out.println("MetaPlan");

        int runMode = 1; // Colateral effects on
        this.runReasoningEngine(runMode);
    }
}

```

## B.2 Código-fonte da Classe Softgoal

Essa seção apresenta o código-fonte da classe Softgoal, pertencente ao pacote `istar.metamodel`. Essa classe é uma classe abstrata que modela uma meta flexível, servindo de raiz para a hierarquia de classes dos diferentes tipos de metas flexíveis: metas flexíveis folhas ou decompostas.

Especificamente, a classe Softgoal define dois métodos abstratos, `getMomentDefuzzify()` e `getMomentDefuzzifyForAlternative(int alternative)`, que devem ser implementados nas classes filhas. Metas flexíveis folhas defuzzificam seu valor calculando o momento das áreas de todos os impactos recebidos. Diferentemente, as metas flexíveis decompostas precisam consultar suas metas flexíveis filhas para retornar um valor.

```
package istar.metamodel;
```

```
/**
 * @author Maurício Serrano

```

```

*/

public abstract class Softgoal {
    private String softgoalName;
    private double weight;

    public Softgoal(String softgoalName) {
        this.softgoalName = softgoalName;
        this.weight = 1.0;
    }

    public Softgoal(String softgoalName, double weight) {
        this.softgoalName = softgoalName;
        this.weight = weight;
    }

    public String getSoftgoalName() {
        return softgoalName;
    }

    public double getWeight() {
        return weight;
    }

    public void setWeight(double weight) {
        this.weight = weight;
    }

    public abstract double getMomentDefuzzify();

    public double getValue() {
        return this.getMomentDefuzzify();
    }

    public abstract double getMomentDefuzzifyForAlternative(int alternative);

    public double getValueForAlternative(int alternative) {
        return this.getMomentDefuzzifyForAlternative(alternative);
    }
}

```

### B.3 Código-fonte da Classe LeafSoftgoal

Essa seção apresenta o código-fonte da classe LeafSoftgoal, que estende a classe Softgoal e pertence ao pacote istar.metamodel. O objetivo dessa classe é especializar metas flexíveis em metas flexíveis folhas, ou seja, metas flexíveis que

não possuem metas flexíveis filhas através de decomposições AND ou OR. Por estender a classe abstrata Softgoal, a classe LeafSoftgoal implementa os métodos getMomentDefuzzify() e getMomentDefuzzify(int alternative). Assim, a defuzzificação por momento é realizada através do cálculo do momento das áreas dos impactos recebidos pela meta flexível.

A importação de biblioteca “import nrc.fuzzy.\*;” faz uma referência à API JAVA disponibilizada no National Research Council, do Canadá.

```
package istar.metamodel;

import nrc.fuzzy.*;

/**
 * @author Maurício Serrano
 */

public class LeafSoftgoal extends Softgoal {

    private FuzzyVariable fuzzyVariable;
    private FuzzyValueVector impacts = new FuzzyValueVector(2);
    private FuzzyValueVector[] alternativesImpacts;
    private FuzzyValueVector[] alternativesNewImpacts;

    public LeafSoftgoal(String softgoalName) {
        super(softgoalName, 1.0);

        try {
            FuzzyVariable fuzzyVariable1 = new FuzzyVariable(softgoalName, -
100.0, 100.0, "percentage");

            fuzzyVariable1.addTerm("denied", new TrapezoidFuzzySet(-100.0, -
100.0, -95.0, -55.0));
            fuzzyVariable1.addTerm("partially_denied", new TrapezoidFuzzySet(-
95.0, -55.0, -45.0, -5.0));
            fuzzyVariable1.addTerm("undecided", new TrapezoidFuzzySet(-45.0, -
5.0, 5.0, 45.0));
            fuzzyVariable1.addTerm("partially_satisfied", new
TrapezoidFuzzySet(5.0, 45.0, 55.0, 95.0));
            fuzzyVariable1.addTerm("satisfied", new TrapezoidFuzzySet(55.0, 95.0,
100.0, 100.0));

            this.fuzzyVariable = fuzzyVariable1;

        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }
}
```

```

public LeafSoftgoal(String softgoalName, double weight) {
    super(softgoalName, weight);

    try {
        FuzzyVariable fuzzyVariable1 = new FuzzyVariable(softgoalName, -
100.0, 100.0, "percentage");

        fuzzyVariable1.addTerm("denied", new TrapezoidFuzzySet(-100.0, -
100.0, -95.0, -55.0));
        fuzzyVariable1.addTerm("partially_denied", new TrapezoidFuzzySet(-
95.0, -55.0, -45.0, -5.0));
        fuzzyVariable1.addTerm("undecided", new TrapezoidFuzzySet(-45.0, -
5.0, 5.0, 45.0));
        fuzzyVariable1.addTerm("partially_satisfied", new
TrapezoidFuzzySet(5.0, 45.0, 55.0, 95.0));
        fuzzyVariable1.addTerm("satisfied", new TrapezoidFuzzySet(55.0, 95.0,
100.0, 100.0));

        this.fuzzyVariable = fuzzyVariable1;

    } catch (Exception ex) {
        ex.printStackTrace();
    }
}

public FuzzyVariable getFuzzyVariable() {
    return this.fuzzyVariable;
}

public FuzzyValueVector getImpacts() {
    return this.impacts;
}

public void setImpacts(FuzzyValueVector impacts) {

    if (impacts == null) this.impacts = null;
    if (impacts != null && impacts.isEmpty()) this.impacts = impacts;
    if (impacts != null && !impacts.isEmpty()) {
        String fuzzyVariableName = this.fuzzyVariable.getName();
        boolean flag = true;
        for(int i=0; i<impacts.size(); i++) {
            FuzzyValue impact = impacts.fuzzyValueAt(i);
            FuzzyVariable impactFuzzyVariable = impact.getFuzzyVariable();
            String impactFuzzyVariableName = impactFuzzyVariable.getName();
            if (!impactFuzzyVariableName.equalsIgnoreCase(fuzzyVariableName))
                flag = false;
        }
        if (flag) this.impacts = impacts;
    }
}

```

```

}

public int getAlternativesImpactsSize() {
    if (this.alternativesImpacts == null) {
        return -1;
    } else {
        return this.alternativesImpacts.length;
    }
}

public void setAlternativeImpactsSize(int size) {

    this.alternativesImpacts = new FuzzyValueVector[size];
    for(int i=0; i<size; i++) {
        this.alternativesImpacts[i] = new
FuzzyValueVector(this.impacts.size()+1);
    }
    for(int i=0; i<this.impacts.size(); i++) {
        FuzzyValue impact = this.impacts.fuzzyValueAt(i);
        for(int j=0; j<size; j++) {
            this.alternativesImpacts[j].addFuzzyValue(impact);
        }
    }

    this.alternativesNewImpacts = new FuzzyValueVector[size];
    for(int i=0; i<size; i++) {
        this.alternativesNewImpacts[i] = new FuzzyValueVector(1);
    }
}

public void addImpactToAlternative(FuzzyValue fuzzyValue, int alternative)
{

    String fuzzyVariableName = this.fuzzyVariable.getName();
    String impactFuzzyVariableName =
fuzzyValue.getFuzzyVariable().getName();

    if (impactFuzzyVariableName.equalsIgnoreCase(fuzzyVariableName)) {
        if (alternative >=0 && alternative < this.getAlternativesImpactsSize()) {
            this.alternativesImpacts[alternative].addFuzzyValue(fuzzyValue);
            this.alternativesNewImpacts[alternative].addFuzzyValue(fuzzyValue);
        }
    }
}

public void applyAlternative(int alternative) {

    if (alternative >=0 && alternative < this.getAlternativesImpactsSize()) {
        //this.setImpacts(this.alternativesImpacts[alternative]);
        this.impacts.concat(this.alternativesNewImpacts[alternative]);
    }
}

```



```

        this.alternativesImpacts = null;
        this.alternativesNewImpacts = null;
    }
}

public FuzzyValueVector getAlternativeImpacts(int alternative) {

    if (alternative >=0 && alternative < this.getAlternativesImpactsSize()) {
        return this.alternativesImpacts[alternative];
    } else {
        return null;
    }
}

public FuzzyValueVector getAlternativeNewImpacts(int alternative) {

    if (alternative >=0 && alternative < this.getAlternativesImpactsSize()) {
        return this.alternativesNewImpacts[alternative];
    } else {
        return null;
    }
}

@Override
public double getMomentDefuzzify() {
    try {
        return this.impacts.momentDefuzzify();
    } catch (Exception ex) {
        ex.printStackTrace();
        return 0.0;
    }
}

@Override
public double getMomentDefuzzifyForAlternative(int alternative) {
    try {
        if (alternative >=0 && alternative < this.getAlternativesImpactsSize()) {
            return this.alternativesImpacts[alternative].momentDefuzzify();
        } else {
            if (this.impacts.isEmpty()) {
                return 0.0;
            } else {
                return this.impacts.momentDefuzzify();
            }
        }
    } catch (Exception ex) {
        ex.printStackTrace();
        return 0.0;
    }
}

```

```

    }
}

```

#### B.4 Código-fonte da Classe DecomposedSoftgoal

Essa seção apresenta o código-fonte da classe DecomposedSoftgoal, que estende a classe Softgoal e pertence ao pacote istar.metamodel. O objetivo dessa classe é especializar metas flexíveis em metas flexíveis decompostas, ou seja, metas flexíveis que possuem metas flexíveis filhas através de decomposições AND ou OR. Por estender a classe abstrata Softgoal, a classe LeafSoftgoal implementa os métodos getMomentDefuzzify() e getMomentDefuzzify(int alternative). Assim, a defuzzificação por momento é realizada através de acessos aos objetos das metas flexíveis filhas e de acordo com o tipo de decomposição.

```

package istar.metamodel;

/**
 * @author Maurício Serrano
 */

public class DecomposedSoftgoal extends Softgoal {

    private DecompositionType decompositionType;
    private Softgoal[] children;

    public DecomposedSoftgoal(String softgoal, DecompositionType
    decompositionType, Softgoal[] children) {
        super(softgoal, 1.0);
        this.decompositionType = decompositionType;
        this.children = children;
    }

    public DecomposedSoftgoal(String softgoal, double weight,
    DecompositionType decompositionType, Softgoal[] children) {
        super(softgoal, weight);
        this.decompositionType = decompositionType;
        this.children = children;
    }

    @Override
    public double getMomentDefuzzify() {
        //throw new UnsupportedOperationException("Not supported yet.");
        double bigger = -100.0;
        double smaller = 100.0;
    }
}

```

```

for(int i=0; i<this.children.length; i++) {
    Softgoal softgoal = this.children[i];
    double value = softgoal.getValue();
    if (value > bigger) {
        bigger = value;
    }
    if (value < smaller) {
        smaller = value;
    }
}

switch(decompositionType) {
    case AND:
        return smaller;
    case OR:
        return bigger;
    default:
        return 0.0;
}
}

@Override
public double getMomentDefuzzifyForAlternative(int alternative) {
    double bigger = -100.0;
    double smaller = 100.0;

    for(int i=0; i<this.children.length; i++) {
        Softgoal softgoal = this.children[i];
        double value = softgoal.getValueForAlternative(alternative);
        if (value > bigger) {
            bigger = value;
        }
        if (value < smaller) {
            smaller = value;
        }
    }

    switch(decompositionType) {
        case AND:
            return smaller;
        case OR:
            return bigger;
        default:
            return 0.0;
    }
}
}
}

```

## B.5 Código-fonte da Enumeração DecompositionType

Essa seção apresenta o código-fonte da enumeração DecompositionType. Essa enumeração pertence ao pacote `istar.metamodel` e define constantes para os tipos de decomposições de metas flexíveis aceitas pela máquina de raciocínio nebulosa.

```
package istar.metamodel;

/**
 * @author Maurício Serrano
 */

public enum DecompositionType {
    AND ("and"),
    OR ("or");

    private final String label; //in lowercase

    DecompositionType(String label) {
        this.label = label;
    }

    public String getLabel() {
        return this.label;
    }
}
```

## B.6 Código-fonte da Classe ImpactSoftgoal

Essa seção apresenta o código-fonte da classe ImpactSoftgoal, que estende a classe Plan do *framework* JADEx e pertence ao pacote `istar.metamodel`. Para o entendimento dessa classe é necessário compreender o mecanismo de sincronização de estados das metas flexíveis entre os agentes (Capítulo 4, Seção 4.2).

A classe ImpactSoftgoal é um plano para agentes executado após o recebimento de uma mensagem de sincronização de estados de metas flexíveis. A mensagem contém o nome de uma meta flexível e os impactos que a mesma deve receber. Assim, essa classe efetua os impactos especificados na mensagem, sendo utilizada por todos os agentes intencionais que lidam com metas flexíveis.

A importação de biblioteca “import nrc.fuzzy.\*;” faz uma referência à API JAVA disponibilizada no National Research Council, do Canadá.

```

package istar.metamodel;

import jadex.runtime.IMessageEvent;
import jadex.runtime.Plan;
import nrc.fuzzy.*;

/**
 * @author Maurício Serrano
 */

public class ImpactSoftgoal extends Plan {

    @Override
    public void body() {

        IMessageEvent message = (IMessageEvent) getInitialEvent();
        SoftgoalImpacts softgoalImpacts = (SoftgoalImpacts) message.getContent();

        Softgoal[] softgoals = (Softgoal[])
this.getBeliefbase().getBeliefSet("softgoals").getFacts();

        for (int i=0; i< softgoals.length; i++) {
            Softgoal tempSoftgoal = softgoals[i];

            if
(tempSoftgoal.getSoftgoalName().equalsIgnoreCase(softgoalImpacts.getSoftgo
alName())) {

                LeafSoftgoal tempLeafSoftgoal = (LeafSoftgoal) tempSoftgoal;
                String[] impacts = softgoalImpacts.getImpacts();
                FuzzyValue impact;
                FuzzyVariable fuzzyVariable = tempLeafSoftgoal.getFuzzyVariable();

                for (int j=0; j<impacts.length; j++) {
                    try {
                        String tempImpact = impacts[j];
                        impact = new FuzzyValue(fuzzyVariable, tempImpact);
                        //System.out.println(impact.plotFuzzyValue("+"));
                        impact.setLinguisticExpression(tempImpact);

                        tempLeafSoftgoal.getImpacts().addFuzzyValue(impact);

                    } catch (InvalidLinguisticExpressionException ex) {
                        ex.printStackTrace();
                    }
                }
            }
        }
    }
}

```

```

        String string = "Softgoal " + softgoalImpacts.getSoftgoalName();
        string = string + " impacted by ";
        string = string + softgoalImpacts.getImpacts().length + " impacts.";
        System.out.println(string);
        break;
    }
}
}
}
}

```

## B.7 Código-fonte da Classe Task

Essa seção apresenta o código-fonte da classe Task, pertencente ao pacote `istar.metamodel`. A classe Task modela uma tarefa de um modelo `i*` e suas contribuições para as metas flexíveis. Essa classe é necessária, pois planos de agentes intencionais JADEx não modelam contribuições para metas flexíveis, acarretando uma perda de informação quando abaixa-se o nível de abstração, impossibilitando o agente de raciocinar nesses termos.

A classe Task também possui um atributo `FuzzyValue fuzzyValue`, que representa o grau de satisfação com que a tarefa foi executada. A importação de biblioteca “`import nrc.fuzzy.*;`” faz uma referência à API JAVA disponibilizada no National Research Council, do Canadá.

```

package istar.metamodel;

import nrc.fuzzy.*;

/**
 * @author Maurício Serrano
 */

public class Task {

    private String taskName;
    private Contribution[] softgoalContributions;
    private FuzzyValue fuzzyValue;

    public Task() {
        this.taskName = "";
        this.softgoalContributions = null;
    }
}

```

```

    this.fuzzyValue = null;
}

public Task(String taskName) {
    this.taskName = taskName;
    try {
        //RightLinearFunction rlf = new RightLinearFunction();
        //LeftLinearFunction llf = new LeftLinearFunction();

        FuzzyVariable fuzzyVariable = new FuzzyVariable(taskName, -100.0,
100.0, "percentage");
        fuzzyVariable.addTerm("denied", new TrapezoidFuzzySet(-100.0, -100.0,
-95.0, -55.0));
        fuzzyVariable.addTerm("partially_denied", new TrapezoidFuzzySet(-95.0,
-55.0, -45.0, -5.0));
        fuzzyVariable.addTerm("undecided", new TrapezoidFuzzySet(-45.0, -5.0,
5.0, 45.0));
        fuzzyVariable.addTerm("partially_satisfied", new
TrapezoidFuzzySet(5.0, 45.0, 55.0, 95.0));
        fuzzyVariable.addTerm("satisfied", new TrapezoidFuzzySet(55.0, 95.0,
100.0, 100.0));

        this.fuzzyValue = new FuzzyValue(fuzzyVariable, "undecided");
    } catch (Exception ex) {
        ex.printStackTrace();
    }
}

public String getTaskName() {
    return taskName;
}

public void setTaskName(String taskName) {
    this.taskName = taskName;
}

public Contribution[] getSoftgoalContributions() {
    return softgoalContributions;
}

public void setSoftgoalContributions(Contribution[] softgoalContributions) {
    this.softgoalContributions = softgoalContributions;
}

public FuzzyValue getFuzzyValue() {
    return fuzzyValue;
}

public void setFuzzyValue(FuzzyValue fuzzyValue) {
    this.fuzzyValue = fuzzyValue;
}

```

```

    }

    public FuzzyVariable getFuzzyVariable() {
        return this.fuzzyValue.getFuzzyVariable();
    }
}

```

## B.8 Código-fonte da Classe Contribution

Essa seção apresenta o código fonte da classe Contribution, pertencente ao pacote `istar.metamodel`. Essa classe modela um elo de contribuição do *framework* `i*`, onde temos: o *contributor*, ou seja, o elemento do modelo que é a origem da contribuição; o tipo de contribuição, como *make*, *help*, entre outros; e o *contributed*, ou seja, o elemento do modelo que recebe a contribuição.

```

package istar.metamodel;

/**
 * @author Maurício Serrano
 */

public class Contribution {

    private String contributor;
    private ContributionType contributionType;
    private String contributed;

    public Contribution() {
        this.contributor = "";
        this.contributionType = ContributionType.UNKNOWN;
        this.contributed = "";
    }

    public Contribution(String contributor, ContributionType contributionType,
        String contributed) {
        this.contributor = contributor;
        this.contributionType = contributionType;
        this.contributed = contributed;
    }

    public String getContributor() {
        return contributor;
    }
}

```



```

public void setContributor(String contributor) {
    this.contributor = contributor;
}

public ContributionType getContributionType() {
    return contributionType;
}

public void setContributionType(ContributionType contributionType) {
    this.contributionType = contributionType;
}

public String getContributed() {
    return contributed;
}

public void setContributed(String contributed) {
    this.contributed = contributed;
}
}

```

## B.9 Código-fonte da Enumeração ContributionType

Essa seção apresenta o código-fonte da enumeração ContributionType, pertencente ao pacote istar.metamodel. Essa enumeração define constantes para os tipos de contribuição aceitos pela máquina de raciocínio nebulosa.

```

package istar.metamodel;

/**
 * @author Maurício Serrano
 */

public enum ContributionType {
    MAKE ("make"),
    HELP ("help"),
    HURT ("hurt"),
    BREAK ("break"),
    SOMEPLUS ("someplus"),
    SOMEMINUS ("someminus"),
    AND ("and"),
    OR ("or"),
    UNKNOWN ("unknown");

    private final String label; //in lowercase

```

```
ContributionType(String label) {  
    this.label = label;  
}  
  
public String getLabel() {  
    return this.label;  
}  
  
}
```

## **Apêndice C. Código-Fonte das Principais Situações do Estudo de Caso Lattes-Scholar**

*Nesse apêndice apresentamos o código-fonte das principais situações do estudo de caso Lattes-Scholar. O código apresentado inclui Agent Definition Files (ADFs) em XML para agentes e capacidades e código em JAVA para os planos. O código JAVA foi desenvolvido programando-se abaixo da descrição dos episódios dos cenários.*

### **C.1 Situação “Buscar o Pesquisador”**

Esta seção apresenta o código-fonte dos artefatos envolvidos na situação “Buscar o pesquisador”. Essa situação ocorre quando o agente Gerente do SMA recebe da página *Web* do Lattes-Scholar o nome de um pesquisador informado pelo cidadão e faz uma requisição ao agente Buscador. Nessa situação, o agente Buscador utiliza a capacidade Google para pesquisar no Google diversos currículos Lattes de pesquisadores que possuem o nome semelhante ao informado pelo cidadão. Uma URL de busca no Google é construída com base no nome informado e são extraídas da página de resposta os dados dos pesquisadores.

#### **C.1.1 Código-fonte do ADF do Agente Buscador `buscador.agent.xml`**

Esta seção apresenta o código-fonte do ADF em XML do agente Buscador. Os principais trechos apresentados são a declaração da capacidade Google, a meta “`pesquisadores_sejam_encontrados`” delegada pelo agente Buscador à capacidade Google e a tarefa “`execute_rp_request`”, disparada quando o agente Buscador recebe uma requisição do agente Gerente do SMA.

<!--

Buscador Agent.

-->

```

<agent xmlns="http://jadex.sourceforge.net/jadex"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://jadex.sourceforge.net/jadex
    http://jadex.sourceforge.net/jadex-0.96.xsd"
  name="Buscador"
  package="lattescholar.sma.buscador">

  <imports>
  <import>jadex.util.*</import>
  <import>jadex.runtime.*</import>
  <import>jadex.adapter.fipa.*</import>
  <import>istar.metamodel.*</import>
  </imports>

  <capabilities>
  <capability name="google" file="lattescholar.sma.google.Google"/>

  <capability name="dfcap" file="jadex.planlib.DF"/>
  <capability name="procap" file="jadex.planlib.Protocols"/>
  <capability name="ReasoningEngine"
    file="istar.metamodel.PropagationSimulator"/>
  </capabilities>

  <beliefs>
  ...
  </beliefs>

  <goals>
  <achievegoalref name="pesquisadores_sejam_encontrados">
    <concrete ref="google.pesquisadores_sejam_encontrados"/>
  </achievegoalref>

  ...

  <!-- (Participant side) Request protocol interaction goal (top-level goal). -->

```

```

<performgoalref name="rp_receiver_interaction">
  <concrete ref="procap.rp_receiver_interaction"/>
</performgoalref>

<!-- (Participant side) Decide upon a requested task will be executed. -->
<querygoalref name="rp_request_be_decided">
  <concrete ref="procap.rp_decide_request"/>
</querygoalref>

<!-- (Participant side) Execute the requested task. -->
<achievegoalref name="rp_request_be_executed">
  <concrete ref="procap.rp_execute_request"/>
</achievegoalref>
</goals>

<plans>

...

<plan name="decide_rp_request">
  <parameter name="initiator" class="AgentIdentifier">
    <goalmapping ref="rp_request_be_decided.initiator"/>
  </parameter>
  <parameter name="action" class="Object">
    <goalmapping ref="rp_request_be_decided.action"/>
  </parameter>
  <parameter name="accept" class="Boolean" direction="out">
    <goalmapping ref="rp_request_be_decided.accept"/>
  </parameter>
  <body class="DecideRPRequest"/>
  <trigger>
    <goal ref="rp_request_be_decided"/>
  </trigger>
</plan>

<plan name="execute_rp_request">
  <parameter name="initiator" class="AgentIdentifier">

```

```

        <goalmapping ref="rp_request_be_executed.initiator"/>
    </parameter>
    <parameter name="action" class="Object">
        <goalmapping ref="rp_request_be_executed.action"/>
    </parameter>
    <parameter name="result" class="Object" direction="out" optional="true">
        <goalmapping ref="rp_request_be_executed.result"/>
    </parameter>
    <body class="ExecuteRPRequest"/>
    <trigger>
        <goal ref="rp_request_be_executed"/>
    </trigger>
</plan>
</plans>

...

</agent>

```

### C.1.2 Código-fonte da Classe ExecuteRPRequest.java

Esta seção apresenta o código-fonte da classe JAVA “ExecuteRPRequest”, do pacote lattescholar.sma.buscador, que representa o plano do agente Buscador disparado quando o agente recebe uma requisição do Gerente do SMA. Em linhas gerais, o agente Buscador recebe do agente Gerente do SMA o nome do pesquisador informado pelo cidadão à página do Lattes-Scholar. A meta "pesquisadores\_sejam\_encontrados" é delegada à capacidade Google e esse nome é repassado como parâmetro da meta. O retorno é uma lista de dados de pesquisadores.

```

package lattescholar.sma.buscador;

import jade.core.AID;

import jadex.adapter.fipa.AgentDescription;
import jadex.adapter.fipa.AgentIdentifier;

```

```

import jadex.adapter.fipa.SFipa;
import jadex.adapter.fipa.SearchConstraints;
import jadex.adapter.fipa.ServiceDescription;
import jadex.runtime.IGoal;
import jadex.runtime.Plan;
import lattescholar.sma.resources.DadosDoPesquisador;

/**
 * @author Maurício Serrano
 */
public class ExecuteRPRequest extends Plan {

    private String content;
    private String buscaNoGoogle;
    private DadosDoPesquisador[] dadosDosPesquisadores;

    public ExecuteRPRequest() {
    }

    @Override
    public void body() {

        System.out.println("Buscador agent running ExecuteRPRequest plan.");

        Object action = this.getParameter("action").getValue();

        AgentIdentifier gerente = (AgentIdentifier) this.getParameter("initiator").getValue();
        AID gerenteAID = new AID(gerente.getName(), true);

        if (action instanceof String) {
            this.content = (String) action;
            if (content.startsWith("<nome")) {

                //Título: Recuperar as urls dos currículos dos pesquisadores
                //Atores: Buscador, Gerente do SMA, Google, Repositório de Currículos
                //Objetivo: Dados Dos Pesquisadores sejam encontrados
                //Recursos: Nome do pesquisador, dados dos pesquisadores (nome, url da foto,

```

```

//      url do currículo), número de pesquisas realizadas
//Episódios:

//Episódio 01-Recuperar as URLs dos currículos dos pesquisadores

this.RecuperarAsUrlsDosCurriculosDosPesquisadores();

//Episódio 02-Recuperar os nomes completos e as fotos dos pesquisadores

this.RecuperarOsNomesCompletoSEAsFotosDosPesquisadores();

//Episódio 03-Enviar para o Gerente do SMA os dados dos pesquisadores

Object[] result = new Object[2];
result[0] = buscaNoGoogle;
result[1] = dadosDosPesquisadores;
this.getParameter("result").setValue(result);
} else {
    this.fail();
}
} else {
    this.fail();
}
}
}

public void RecuperarAsUrlsDosCurriculosDosPesquisadores() {
    //Título: Recuperar as urls dos currículos dos pesquisadores
    //Atores: Buscador, Gerente do SMA, Google
    //Objetivo: Pesquisadores sejam encontrados
    //Recursos: Nome do pesquisador, dados dos pesquisadores (url do currículo),
    //      número de pesquisas realizadas
    //Episódios:

    //Episódio 01-Receber do Gerente do SMA o nome do pesquisador

    int beginIndex = 21;
    int endIndex = content.indexOf("</nome_do_pesquisador>");

```



```

String nomeDoPesquisador = content.substring(beginIndex, endIndex);

//Episódio 02-Delegar a meta "Pesquisadores sejam encontrados" para a
//      capacidade Google

IGoal PesquisadoresSejamEncontrados =
    this.createGoal("pesquisadores_sejam_encontrados");
PesquisadoresSejamEncontrados.getParameter("nome_do_pesquisador")
    .setValue(nomeDoPesquisador);
this.dispatchSubgoalAndWait(PesquisadoresSejamEncontrados);

//Episódio 03-Receber da capacidade Google as urls dos currículos
//dos pesquisadores

buscaNoGoogle = (String)
    PesquisadoresSejamEncontrados.getParameter("busca_no_google").getValue();
dadosDosPesquisadores = (DadosDoPesquisador[]) PesquisadoresSejamEncontrados
    .getParameter("dados_dos_pesquisadores").getValue();

//Episódio 04-Incrementar em um o número de pesquisas realizadas

Integer searches = (Integer) this.getBeliefbase().getBelief("searches").getFact();
this.getBeliefbase().getBelief("searches").setFact(searches + 1);
}

public void RecuperarOsNomesCompletoSEAsFotosDosPesquisadores() {
    ...
}
}

```

### C.1.3

#### Código-fonte do ADF da Capacidade Google Google.capability.xml

Esta seção apresenta o código-fonte do ADF em XML da capacidade Google. A capacidade Google é utilizada pelo agente Buscador para realizar uma

pesquisa no Google para buscar currículos Lattes de pesquisadores que possuem o nome semelhante ao nome informado pelo cidadão na página do Lattes-Scholar. Os principais trechos apresentados são a meta “pesquisadores\_sejam\_encontrados” e o plano “BuscarPesquisadoresNoGoogle”.

```

<!--
    Google Capability.
-->
<capability xmlns="http://jadex.sourceforge.net/jadex"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://jadex.sourceforge.net/jadex
    http://jadex.sourceforge.net/jadex-0.96.xsd"
  name="Google"
  package="lattescholar.sma.google">

  <imports>
  <import>jadex.util.*</import>
  <import>jadex.runtime.*</import>
  <import>jadex.adapter.fipa.*</import>
  <import>istar.metamodel.*</import>
  </imports>

  <capabilities>
  </capabilities>

  <beliefs>
    ...
  </beliefs>

  <goals>
  <achievegoal name="pesquisadores_sejam_encontrados" exported="true">
    <parameter name="nome_do_pesquisador" class="String"/>
    <parameter name="busca_no_google" class="String" direction="out"/>
    <parameter name="dados_dos_pesquisadores" class="Object[]"
      direction="out"/>
  </achievegoal>

```

```

</goals>

<plans>
  <plan name="buscar_pesquisadores_no_google">
    <parameter name="nome_do_pesquisador" class="String">
      <goalmapping
        ref="pesquisadores_sejam_encontrados.nome_do_pesquisador"/>
      </parameter>
    <parameter name="busca_no_google" class="String" direction="out">
      <goalmapping ref="pesquisadores_sejam_encontrados.busca_no_google"/>
    </parameter>
    <parameter name="dados_dos_pesquisadores" class="Object[]"
      direction="out">
      <goalmapping
        ref="pesquisadores_sejam_encontrados.dados_dos_pesquisadores"/>
    </parameter>
    <body class="BuscarPesquisadoresNoGoogle"/>
    <trigger>
      <goal ref="pesquisadores_sejam_encontrados"/>
    </trigger>
  </plan>
  ...
</plans>
  ...

</capability>

```

#### C.1.4 Código-fonte da Classe `BuscarPesquisadoresNoGoogle.java`

Esta seção apresenta o código-fonte da classe `BuscarPesquisadoresNoGoogle.java`, que implementa o plano de mesmo nome da capacidade Google. Esse plano é disparado quando a capacidade é acionada pelo agente `Buscador`. O plano recebe apenas um parâmetro de entrada, o nome do pesquisador, faz a pesquisa no Google e extrai as URLs dos currículos dos

pesquisadores. Ao final, retorna dois parâmetros de saída: a busca no google e os dados dos pesquisadores.

```

package lattesscholar.sma.google;

import jade.util.leap.HashMap;
import jadex.runtime.Plan;
import java.io.BufferedInputStream;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.URL;
import java.net.URLConnection;
import java.util.logging.Level;
import java.util.logging.Logger;
import lattesscholar.sma.resources.DadosDoPesquisador;
import lattesscholar.sma.resources.StringFilter;

/**
 * @author Maurício Serrano
 */
public class BuscarPesquisadoresNoGoogle extends Plan {

    private String buscaNoGoogle;

    @Override
    public void body() {
        System.out.println("Google capability running BuscarPesquisadoresNoGoogle
            plan.");

        Boolean offlineMode = (Boolean) this.getBeliefbase()
            .getBelief("offline_mode").getFact();
        StringFilter stringFilter = new StringFilter();
    }
}

```

```

//Título: Buscar os pesquisadores no Google
//Atores: Google, Buscador
//Objetivo: Pesquisadores sejam encontrados
//Recursos: nome do pesquisador, dados dos pesquisadores (nome, url da foto,
//      url do currículo, expressão de busca, página de busca do Google,
//      trecho em html de um currículo de um pesquisador
//Episódios:

//Episódio 01-Receber do Buscador o nome do pesquisador

String nomeDoPesquisador = (String)
    this.getParameter("nome_do_pesquisador").getValue();

//Episódio 02-Montar a url de busca com a expressão de busca e o nome do
pesquisador

String content = "";

// modo offline para testes
if (offlineMode) {
    String filePath = "C:\\teste1.html";

    byte[] buffer = new byte[(int) new File(filePath).length()];
    BufferedInputStream f = null;

    try {
        try {
            f = new BufferedInputStream(new FileInputStream(filePath));
            f.read(buffer);
        } catch (Exception ex) {

Logger.getLogger(BuscarPesquisadoresNoGoogle.class.getName()).log(Level.SEVERE,
null, ex);
        }
    } finally {

```

```

if (f != null) {
    try {
        f.close();
    } catch (IOException ignored) {
    }
}
}
}

```

```

buscaNoGoogle = "http://www.google.com.br/search?q=Author" +
    "+site%3Abuscatextual.cnpq.br&hl=pt-BR&biw=1280&bih=681" +
    "&num=20&lr=&ft=i&cr=&safe=images";
content = new String(buffer);

```

```

} else {

```

```

//Episódio 03-Abrir uma conexão com o Google br

```

```

String prefixo = "http://www.google.com.br/search?q=";
String sufixo = "+site%3Abuscatextual.cnpq.br&hl=pt-BR&biw=1280" +
    "&bih=681&num=20&lr=&ft=i&cr=&safe=images";

```

```

URL urlDaBuscaNoGoogle;

```

```

try {
    this.buscaNoGoogle = prefixo + nomeDoPesquisador + sufixo;
    urlDaBuscaNoGoogle = new URL(buscaNoGoogle);

```

```

URLConnection urlConn = urlDaBuscaNoGoogle.openConnection();
urlConn.addRequestProperty("User-Agent", "Mozilla/4.0 (compatible; MSIE
6.0; Windows NT 5.1)");

```

```

//String contentType = urlConn.getContentType();
//System.out.println("contentType:" + contentType);

```

```

InputStream inputStream = urlConn.getInputStream();
InputStreamReader inputStreamReader = new

```

```

        InputStreamReader(inputStream);
        BufferedReader bufferedReader = new BufferedReader(inputStreamReader);
        StringBuilder stringBuilder = new StringBuilder();
        String htmlLine;

```

```

//Episódio 04-Recuperar a página com a busca pelo nome do pesquisador

```

```

while ((htmlLine = bufferedReader.readLine()) != null) {
    stringBuilder.append(htmlLine);
}

```

```

content = stringBuilder.toString();

```

```

} catch (Exception ex) {

```

```

    Logger.getLogger(BuscarPesquisadoresNoGoogle.class.getName()).log(Level.SEVERE,
    null, ex);
}

```

```

//Episódio 05-Extrair da página de busca do Google os trechos

```

```

//em HTML dos currículos dos pesquisadores utilizando o trecho em html

```

```

//de um currículo de um pesquisador

```

```

DadosDoPesquisador[] dadosDosPesquisadores = new DadosDoPesquisador[20];

```

```

int nextUrlSlot = 0;

```

```

HashMap ids = new HashMap();

```

```

for (int i = 0; i < 20; i++) {

```

```

    DadosDoPesquisador dadosDoPesquisador = new DadosDoPesquisador();

```

```

    dadosDoPesquisador.setNome("sem nome");

```

```

    dadosDoPesquisador.setUrlDaFoto("sem foto");

```

```

    dadosDoPesquisador.setUrlDoCurrículo("sem currículo");

```

```

    dadosDosPesquisadores[i] = dadosDoPesquisador;

```

```

}

```

```

do {
    int beginIndex = content.indexOf("<li class=\"g\">") + 37;
    content = content.substring(beginIndex);
    int endIndex = content.indexOf("onmousedown") - 2;
    String url = content.substring(0, endIndex);
    if (url.indexOf("idiomaExibicao=2") < 0) {
        System.out.println(url);
        beginIndex = url.indexOf("id=");
        String id = url.substring(beginIndex + 3);
        endIndex = id.indexOf("&");
        if (endIndex > 0) {
            id = id.substring(0, endIndex);
        }
        System.out.println(id);
        if (!ids.containsKey(id)) {
            ids.put(id, id);
        }
    }

    //Episódio 06-Construir os dados dos pesquisadores

    DadosDoPesquisador tempDadosDoPesquisador;
    tempDadosDoPesquisador = (DadosDoPesquisador)
        dadosDosPesquisadores[nextUrlSlot];
    tempDadosDoPesquisador.setUrlDoCurriculo(stringFilter.escapeAmp(url));
    nextUrlSlot = nextUrlSlot + 1;
}
}
content = content.substring(37);
} while (content.indexOf("<li class=\"g\">") > 0);

//Episódio 07-Enviar os dados dos pesquisadores para o Buscador
this.getParameter("busca_no_google").setValue(buscaNoGoogle);
this.getParameter("dados_dos_pesquisadores").setValue(dadosDosPesquisadores);
}
}

```



## C.2

### Situação “Decidir entre Utilizar o Agente Fornecedor de Citações Local ou um Remoto”

Esta seção apresenta o código-fonte dos artefatos envolvidos na situação “Decidir entre utilizar o agente Fornecedor de Citações local ou um remoto”. Esta situação ocorre quando o agente Gerente do SMA precisa obter as citações das publicações encontradas nas seções do currículo Lattes selecionadas pelo cidadão. Utilizar o agente local contribui positivamente para a meta flexível “Performance”, pois evita requisições a agentes remotos e o tráfego adicional de informações na rede. Entretanto, utilizar o agente local muitas vezes seguidas pode ferir a licença de uso do Google Scholar, que não permite que um mesmo endereço IP realize muitas pesquisas em um curto espaço de tempo. Essa restrição é representada pela meta flexível “Evitar muitas requisições [Google Scholar]”. Utilizar o agente local, portanto, contribui negativamente para essa meta flexível. Utilizar um agente remoto produz contribuições exatamente opostas. Para poder realizar essa tarefa, o agente utiliza a máquina de raciocínio nebulosa para metas flexíveis.

#### C.2.1

##### Código-fonte do ADF do Agente Fornecedor de Citações FornecedorDeCitações.agent.xml

Esta seção apresenta o código-fonte do ADF em XML do agente Fornecedor de Citações. Os principais trechos apresentados referem-se à meta “numero\_de\_citacoes\_da\_publicacao\_seja\_conhecido” e os dois planos que podem atingir essa meta, “RequisitarONumeroDeCitacoesParaOGoogleScholar” e “DelegarAMetaAUmAgenteRemoto”. São apresentados também: o metagoal, disparado pela máquina de deliberação de metas do JADDEX; o metaplano, associado ao metagoal e que dispara a classe MetaPlan, oferecida pela máquina de raciocínio nebulosa para metas flexíveis; o plano CleanOldSearches, que a cada minuto subtrai cinco do número de pesquisas realizadas, permitindo que novas pesquisas sejam feitas; o plano SetupAgent, responsável por inicializar as metas flexíveis, os elos de contribuições e as tarefas; e a configuração default, que dispara o plano SetupAgent.

```

<!--
    FornecedorDeCitacoes Agent.
-->
<agent xmlns="http://jadex.sourceforge.net/jadex"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://jadex.sourceforge.net/jadex
                           http://jadex.sourceforge.net/jadex-0.96.xsd"
       name="FornecedorDeCitacoes"
       package="lattescholar.sma.fornecedordecitacoes">

    <imports>
    <import>jadex.util.*</import>
    <import>jadex.runtime.*</import>
    <import>jadex.adapter.fipa.*</import>
    <import>istar.metamodel.*</import>
    <import>lattescholar.sma.resources.*</import>
    </imports>

    <capabilities>
    <capability name="scholar" file="lattescholar.sma.scholar.Scholar"/>

    <capability name="dfcap" file="jadex.planlib.DF"/>
    <capability name="procap" file="jadex.planlib.Protocols"/>
    <capability name="ReasoningEngine"
               file="istar.metamodel.PropagationSimulator"/>
    </capabilities>

    <beliefs>
    ...

    <beliefset name="tasks" class="Task"/>
    <beliefset name="contributions" class="Contribution"/>
    <beliefset name="softgoals" class="Softgoal"/>
    </beliefs>

    <goals>

```

...

```

<achievegoal name="numero_de_citacoes_da_publicacao_seja_conhecido">
  <parameter name="publicacoes_sem_as_citacoes" class="Publicacao[]"/>
  <parameter name="publicacoes_com_as_citacoes" class="Publicacao[]"
    direction="out"/>
</achievegoal>

```

```

<metagoal name="metagoal1" recalculate="false">
  <parameterset name="applicables" class="ICandidateInfo"/>
  <parameterset name="result" class="ICandidateInfo" direction="out"/>
  <trigger>
    <goal ref="numero_de_citacoes_da_publicacao_seja_conhecido"/>
  </trigger>
</metagoal>

```

...

```

</goals>

```

```

<plans>
  <plan name="requisitar_o_numero_de_citacoes_para_o_google_scholar">
    <parameter name="publicacoes_sem_as_citacoes" class="Publicacao[]">
      <goalmapping ref="numero_de_citacoes_da_publicacao_seja_conhecido
        .publicacoes_sem_as_citacoes"/>
    </parameter>
    <parameter name="publicacoes_com_as_citacoes" class="Publicacao[]"
      direction="out">
      <goalmapping ref="numero_de_citacoes_da_publicacao_seja_conhecido
        .publicacoes_com_as_citacoes"/>
    </parameter>
    <body class="RequisitarONumeroDeCitacoesParaOGoogleScholar "/>
    <trigger>
      <goal ref="numero_de_citacoes_da_publicacao_seja_conhecido"/>
    </trigger>
  </plan>

```

```

<plan name="delegar_a_meta_a_um_agente_remoto">
  <parameter name="publicacoes_sem_as_citacoes" class="Publicacao[]">
    <goalmapping ref="numero_de_citacoes_da_publicacao_seja_conhecido
      .publicacoes_sem_as_citacoes"/>
  </parameter>
  <parameter name="publicacoes_com_as_citacoes" class="Publicacao[]"
    direction="out">
    <goalmapping ref="numero_de_citacoes_da_publicacao_seja_conhecido
      .publicacoes_com_as_citacoes"/>
  </parameter>

  <body class="DelegarAMetaAUmAgenteRemoto"/>
  <trigger>
    <goal ref="numero_de_citacoes_da_publicacao_seja_conhecido"/>
  </trigger>
</plan>

<plan name="metaplan1">
  <body class="MetaPlan"/>
  <trigger>
    <goal ref="metagoal1"/>
  </trigger>
</plan>

<plan name="clean_old_searches">
  <body class="CleanOldSearches"/>
</plan>

<plan name="setup_agent">
  <body class="SetupAgent"/>
</plan>
</plans>

<configurations>
  <configuration name="default">
    <beliefs>

```

```

    <initialbelief ref="rp_filter">
      <fact>IFilter.ALWAYS</fact>
    </initialbelief>
    <initialbelief ref="offline_mode">
      <fact>Boolean.FALSE</fact>
    </initialbelief>
  </beliefs>
  <goals>
    <endgoal ref="df_deregister"/>
  </goals>
  <plans>
    <initialplan name="start" ref="setup_agent"/>
    <initialplan name="timer" ref="clean_old_searches"/>
  </plans>
</configuration>
</configurations>
</agent>

```

## C.2.2 Código-fonte da Classe SetupAgent.java

Esta seção apresenta o código-fonte classe SetupAgent.java, que implementa o plano inicial do agente, responsável por inicializar o agente e as crenças “softgoals” e “tasks”, utilizadas pela máquina de raciocínio nebulosa. É nessa classe que são criadas as metas flexíveis, os elos de contribuição e as tarefas. Esse mesmo plano também registra o agente nas páginas amarelas utilizando a propriedade “searches”, utilizada pelos outros agentes para definir qual agente Fornecedor de Citações fez menos requisições ao Google Scholar.

```

package lattesscholar.sma.fornecedordecitacoes;

import lattesscholar.sma.fornecedordecitacoes.*;
import jadex.adapter.fipa.AgentDescription;
import jadex.adapter.fipa.Property;
import jadex.adapter.fipa.SFipa;
import jadex.adapter.fipa.ServiceDescription;
import jadex.runtime.IGoal;

```

```

import jadex.runtime.Plan;
import istar.metamodel.*;

/**
 * @author Maurício Serrano
 */
public class SetupAgent extends Plan {

    @Override
    public void body() {
        System.out.println("FornecedorDeCitacoes agent running SetupAgent plan.");

        this.getBeliefbase().getBelief("searches").setFact(new Integer(0));

        // registro nas páginas amarelas
        ServiceDescription serviceDescription = new
            ServiceDescription("fornecedor_de_citacoes", "scholar", "lattes-scholar");
        Property property = new Property();
        property.setName("searches");
        property.setValue(new Integer(0));
        serviceDescription.addProperty(property);
        AgentDescription agentDescription =
            SFipa.createAgentDescription(null, serviceDescription);

        IGoal dfRegister = this.createGoal("df_register");
        dfRegister.getParameter("description").setValue(agentDescription);
        this.dispatchSubgoalAndWait(dfRegister);

        try {
            LeafSoftgoal performance = new LeafSoftgoal("Performance", 1.0);
            this.getBeliefbase().getBeliefSet("softgoals").addFact(performance);

            LeafSoftgoal evitarMuitasRequisicoes = new
                LeafSoftgoal("Evitar muitas requisições [Google Scholar]", 1.0);
            this.getBeliefbase().getBeliefSet("softgoals").addFact(evitarMuitasRequisicoes);

        } catch (Exception ex) {

```

```

        ex.printStackTrace();
    }

    try {
        // requisitar contribui positivamente para performance
        // e contribui negativamente para evitar muitas requisições
        Task requisitarAoGoogleScholar = new Task("Requisitar o número de citações
para o Google Scholar");
        Contribution[] contr1 = {
            new Contribution(("Requisitar o número de citações para o Google Scholar",
                ContributionType.HELP, "Performance"),
            new Contribution(("Requisitar o número de citações para o Google Scholar",
                ContributionType.HURT, "Evitar muitas requisições [Google Scholar]"));
        requisitarAoGoogleScholar.setSoftgoalContributions(contr1);
        this.getBeliefbase().getBeliefSet("tasks").addFact(requisitarAoGoogleScholar);

        // delegar a meta contribui negativamente para a performance
        // e contribui positivamente para evitar muitas requisições
        Task delegarAMeta =
            new Task("Delegar a meta a um agente remoto");
        Contribution[] contr2 = {
            new Contribution("Delegar a meta a um agente remoto",
                ContributionType.HURT, "Performance"),
            new Contribution("Delegar a meta a um agente remoto",
                ContributionType.HELP, "Evitar muitas requisições [Google Scholar]"));
        delegarAMeta.setSoftgoalContributions(contr2);
        this.getBeliefbase().getBeliefSet("tasks").
            addFact(delegarAMeta);

    } catch (Exception ex) {
        ex.printStackTrace();
    }
}
}

```

### C.2.3 Código-fonte da Classe RequisitarONumeroDeCitacoesParaOGoogleScholar.java

Esta seção apresenta o código-fonte em JAVA da classe `RequisitarONumeroDeCitacoesParaOGoogleScholar.java`, que representa o plano de mesmo nome do agente Fornecedor de Citações. Esse plano é uma das alternativas disponíveis para se tentar atingir a meta “numero\_de\_citacoes\_da\_publicacao\_seja\_conhecido”. Como esse plano representa a ação de utilizar o agente Fornecedor de Citações local, ou seja, o próprio agente, o código do plano é simples, disparando uma meta do agente. Apenas é necessária a correta passagem de parâmetros de entrada e de saída.

```
package lattesscholar.sma.fornecedordecitacoes;

import jadex.runtime.IGoal;
import jadex.runtime.Plan;
import lattesscholar.sma.resources.Publicacao;

/**
 * @author Maurício Serrano
 */
public class RequisitarONumeroDeCitacoesParaOGoogleScholar extends Plan {

    @Override
    public void body() {
        System.out.println("FornecedorDeCitacoes agent running
            RequisitarONumeroDeCitacoesParaOGoogleScholar plan.");

        Publicacao[] publicacoes = (Publicacao[])
            this.getParameter("publicacoes_sem_as_citacoes").getValue();

        IGoal citacoesDasPublicacoesSejamRecuperadas =
            this.createGoal("citacoes_das_publicacoes_sejam_recuperadas");
        citacoesDasPublicacoesSejamRecuperadas.getParameter("publicacoes")
            .setValue(publicacoes);
        this.dispatchSubgoalAndWait(citacoesDasPublicacoesSejamRecuperadas);
    }
}
```



```

Publicacao[] publicacoesComCitacoes = (Publicacao[])
    citacoesDasPublicacoesSejamRecuperadas.getParameter(
        "publicacoes_com_as_citacoes").getValue();

this.getParameter("publicacoes_com_as_citacoes").setValue(publicacoesComCitacoes);
    }
}

```

#### C.2.4 Código-fonte da Classe DelegarAMetaAUmAgenteRemoto.java

Esta seção apresenta o código-fonte em JAVA da classe DelegarAMetaAUmAgenteRemoto.java, que representa a implementação do plano de mesmo nome do agente Fornecedor de Citações. Esse plano é a segunda alternativa disponível para se tentar atingir a meta “numero\_de\_citacoes\_da\_publicacao\_seja\_conhecido”. Para delegar a meta a um agente remoto, o plano consulta o serviço de páginas amarelas da plataforma JADEX e delega a meta ao agente Fornecedor de Citações remoto que menos realizou pesquisas no Google Scholar, efetuando, assim, o balanceamento da carga entre os diversos agentes.

```

package lattesscholar.sma.fornecedordecitacoes;

import jadex.adapter.fipa.AgentDescription;
import jadex.adapter.fipa.AgentIdentifier;
import jadex.adapter.fipa.SFipa;
import jadex.adapter.fipa.SearchConstraints;
import jadex.adapter.fipa.ServiceDescription;
import jadex.runtime.IGoal;
import jadex.runtime.Plan;
import lattesscholar.sma.resources.Publicacao;

/**
 * @author Maurício Serrano
 */
public class DelegarAMetaAUmAgenteRemoto extends Plan {

```

```

@Override
public void body() {
    System.out.println("FornecedorDeCitacoes agent running
        DelegarAMetaAUmAgenteRemoto plan.");

    Publicacao[] publicacoes = (Publicacao[])
        this.getParameter("publicacoes_sem_as_citacoes").getValue();

    IGoal df_search = createGoal("df_search");

    ServiceDescription sd = SFipa.createServiceDescription("fornecedor_de_citacoes",
        null, "lattes-scholar");
    AgentDescription ad = SFipa.createAgentDescription(null, sd);

    df_search.getParameter("description").setValue(ad);
    SearchConstraints sc = new SearchConstraints();
    sc.setMaxResults(20);
    df_search.getParameter("constraints").setValue(sc);

    dispatchSubgoalAndWait(df_search);

    AgentDescription[] result = (AgentDescription[])
        df_search.getParameterSet("result").getValues();

    if (result.length == 0) {
        System.out.println("FornecedorDeCitacoes agent not found in DF.");
        fail();
    }

    AgentIdentifier fornecedorDeCitacoes = null;
    Long lowerSearches = Long.MAX_VALUE;

    for (int i = 0; i < result.length; i++) {
        String property = result[i].getServices()[0].getProperty(0).getName();
        System.out.println("Property: " + property);
        Long auxSearches = (Long) result[i].getServices()[0].getProperty(0).getValue();
    }
}

```

```

    if (auxSearches < lowerSearches) {
        lowerSearches = auxSearches;
        fornecedorDeCitacoes = result[i].getName();
    }
}

```

```

IGoal requestProtocolBeInitiated = this.createGoal("request_protocol_be_initiated");

```

```

requestProtocolBeInitiated.getParameter("receiver").setValue(fornecedorDeCitacoes);
requestProtocolBeInitiated.getParameter("action").setValue(publicacoes);
this.dispatchSubgoalAndWait(requestProtocolBeInitiated);

```

```

Publicacao[] publicacoesComCitacoes = (Publicacao[])
    requestProtocolBeInitiated.getParameter("result").getValue();

```

```

this.getParameter("publicacoes_com_as_citacoes").setValue(publicacoesComCitacoes);
}
}

```

### C.3 Código-fonte para a Situação “Obter as Citações da Publicação no Google Scholar”

Esta seção apresenta o código-fonte para a situação “Obter as citações da publicação no Google Scholar”. Esta situação ocorre quando o agente Gerente do SMA precisa obter as citações das publicações encontradas nas seções do currículo Lattes selecionadas pelo cidadão. Diferentemente da Seção C.2, o foco dessa situação não é a decisão entre utilizar um agente local ou remoto; o foco é o processo realizado para se obter as citações das publicações, como a pesquisa no Google Scholar e a análise das páginas das publicações encontradas. Nessa situação, o agente Fornecedor de Citações utiliza a capacidade Scholar para tentar atingir a meta “citacoes\_das\_publicacoes\_sejam\_recuperadas”, delegada pelo agente Gerente do SMA. Essa tarefa inclui a obtenção da página da publicação no Google Scholar e a análise dessa página para extrair as citações da publicação.

### C.3.1 Código-fonte do ADF do Agente Fornecedor de Citações FornecedorDeCitacoes.agent.xml

Esta seção apresenta o código-fonte do ADF em XML do agente Fornecedor de Citações. Os principais trechos apresentados são a capacidade Scholar, a meta “citacoes\_das\_publicacoes\_sejam\_recuperadas”, delegada para a capacidade Scholar, e o plano “ExecuteRPrequest”, disparado quando o agente Fornecedor de Citações recebe uma requisição do agente Gerente do SMA. Os trechos apresentados não são os mesmos apresentados no Apêndice C.2.1, uma vez que o foco é a contagem das citações da publicação na página do Google Scholar e não o funcionamento da máquina de raciocínio nebulosa.

```
<!--
  FornecedorDeCitacoes Agent.
-->
<agent xmlns="http://jadex.sourceforge.net/jadex"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://jadex.sourceforge.net/jadex
    http://jadex.sourceforge.net/jadex-0.96.xsd"
  name="FornecedorDeCitacoes"
  package="lattescholar.sma.fornecedordecitacoes">

  <imports>
  <import>jadex.util.*</import>
  <import>jadex.runtime.*</import>
  <import>jadex.adapter.fipa.*</import>
  <import>istar.metamodel.*</import>
  <import>lattescholar.sma.resources.*</import>
  </imports>

  <capabilities>
  <capability name="scholar" file="lattescholar.sma.scholar.Scholar"/>
  ...

</capabilities>
```

```

<beliefs>
  ...
</beliefs>

<goals>
  <achievegoalref name="citacoes_das_publicacoes_sejam_recuperadas">
    <concrete ref="scholar.citacoes_das_publicacoes_sejam_recuperadas"/>
  </achievegoalref>

  ...

  <!-- (Participant side) Decide upon a requested task will be executed. -->
  <querygoalref name="rp_request_be_decided">
    <concrete ref="procap.rp_decide_request"/>
  </querygoalref>

  <!-- (Participant side) Execute the requested task. -->
  <achievegoalref name="rp_request_be_executed">
    <concrete ref="procap.rp_execute_request"/>
  </achievegoalref>
</goals>

<plans>

  ...

  <plan name="decide_rp_request">
    <parameter name="initiator" class="AgentIdentifier">
      <goalmapping ref="rp_request_be_decided.initiator"/>
    </parameter>
    <parameter name="action" class="Object">
      <goalmapping ref="rp_request_be_decided.action"/>
    </parameter>
    <parameter name="accept" class="Boolean" direction="out">
      <goalmapping ref="rp_request_be_decided.accept"/>
    </parameter>
  </plan>

```

```

    <body class="DecideRPRequest"/>
    <trigger>
      <goal ref="rp_request_be_decided"/>
    </trigger>
  </plan>

  <plan name="execute_rp_request">
    <parameter name="initiator" class="AgentIdentifier">
      <goalmapping ref="rp_request_be_executed.initiator"/>
    </parameter>
    <parameter name="action" class="Object">
      <goalmapping ref="rp_request_be_executed.action"/>
    </parameter>
    <parameter name="result" class="Object" direction="out" optional="true">
      <goalmapping ref="rp_request_be_executed.result"/>
    </parameter>
    <body class="ExecuteRPRequest"/>
    <trigger>
      <goal ref="rp_request_be_executed"/>
    </trigger>
  </plan>
</plans>

...

</agent>

```

### C.3.2 Código-fonte da Classe ExecuteRPRequest.java

Esta seção apresenta o código-fonte da classe JAVA “ExecuteRPRequest”, do pacote lattescholar.sma.fornecedordecitacoes, que representa o plano do agente Fornecedor de Citações disparado quando o agente recebe uma requisição do Gerente do SMA. Em linhas gerais, o agente Fornecedor de Citações recebe do agente Gerente do SMA uma lista de publicações e obtém as páginas do Google Scholar para cada uma delas. Uma vez obtidas as páginas, a capacidade Scholar analisa-as e extrai o número de citações das publicações.

```

package lattesscholar.sma.fornecedordecitacoes;

import jade.core.AID;
import jadex.adapter.fipa.AgentIdentifier;
import jadex.runtime.IGoal;
import jadex.runtime.Plan;
import lattesscholar.sma.resources.Publicacao;

/**
 * @author Maurício Serrano
 */
public class ExecuteRPRequest extends Plan {

    public ExecuteRPRequest() {

    }

    @Override
    public void body() {

        System.out.println("FornecedorDeCitacoes agent running ExecuteRPRequest
plan.");

        Object action = this.getParameter("action").getValue();

        AgentIdentifier gerente = (AgentIdentifier) this.getParameter("initiator").getValue();
        AID gerenteAID = new AID(gerente.getName(), true);

        if (action instanceof Publicacao[]) {

            //Título: Delegar a meta para a capacidade Scholar
            //Ator: Fornecedor de Citações, Gerente do SMA, Scholar
            //Objetivo: Citações das publicações sejam recuperadas
            //Recursos: lista de publicações,
            //    Número de pesquisas realizadas,
            //    Número de publicações
            //Episódios:

```

```
//Episódio 01-Receber do Gerente do SMA a lista de publicacoes
```

```
Publicacao[] publicacoes = (Publicacao[]) action;
```

```
//Episódio 02-Delegar a meta "Citações das publicações sejam recuperadas"
```

```
// para a capacidade Scholar
```

```
IGoal citacoesDasPublicacoesSejamRecuperadas =
```

```
    this.createGoal("citacoes_das_publicacoes_sejam_recuperadas");
```

```
    citacoesDasPublicacoesSejamRecuperadas.getParameter("publicacoes")
```

```
        .setValue(publicacoes);
```

```
    this.dispatchSubgoalAndWait(citacoesDasPublicacoesSejamRecuperadas);
```

```
//Episódio 03-Receber da capacidade Scholar as publicacoes com as citacoes
```

```
Publicacao[] publicacoesComAsCitacoes = (Publicacao[])
```

```
    citacoesDasPublicacoesSejamRecuperadas.getParameter(
```

```
        "publicacoes_com_as_citacoes").getValue();
```

```
//Episódio 04-Incrementar o número de pesquisas realizadas
```

```
//de acordo com o número de publicacoes
```

```
Integer searches = (Integer) this.getBeliefbase().getBelief("searches").getFact();
```

```
this.getBeliefbase().getBelief("searches").setFact(searches + publicacoes.length);
```

```
//Episódio 05-Enviar para o agente Gerente do SMA as publicações
```

```
//com as citações
```

```
    this.getParameter("result").setValue(publicacoesComAsCitacoes);
```

```
    }
```

```
    else {
```

```
        this.fail();
```

```
    }
```

```
    }
```

```
    }
```



### C.3.3 Código-fonte do ADF da Capacidade Scholar Scholar.capability.xml

Esta seção apresenta o código-fonte do ADF em XML da capacidade Scholar. Os principais trechos apresentados são as metas “citacoes\_das\_publicacoes\_sejam\_recuperadas”, “publicacao\_seja\_pesquisada” e “citacoes\_sejam\_extraidas\_da\_pagina”, os planos ObterAsCitacoesNoGoogleScholar “RecuperarAPaginaDaPublicacaoNoGoogleScholar” e “ParsearAPaginaDaPublicacao”.

```

<!--
    Scholar Capability.
-->
<capability xmlns="http://jadex.sourceforge.net/jadex"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://jadex.sourceforge.net/jadex
        http://jadex.sourceforge.net/jadex-0.96.xsd"
    name="Scholar"
    package="lattescholar.sma.scholar">

    <imports>
    <import>jadex.util.*</import>
    <import>jadex.runtime.*</import>
    <import>jadex.adapter.fipa.*</import>
    <import>istar.metamodel.*</import>
    <import>lattescholar.sma.resources.*</import>
    </imports>

    <capabilities>
    </capabilities>

    <beliefs>
        ....
    </beliefs>

    <goals>
        <achievegoal name="citacoes_das_publicacoes_sejam_recuperadas"

```

```

        exported="true">
        <parameter name="publicacoes" class="Publicacao[]"/>
        <parameter name="publicacoes_com_as_citacoes" class="Publicacao[]"
        direction="out"/>
    </achievegoal>

    <achievegoal name="publicacao_seja_pesquisada" exported="true">
        <parameter name="publicacao" class="Publicacao"/>
        <parameter name="elo" class="String" direction="out"/>
        <parameter name="pagina" class="String" direction="out"/>
    </achievegoal>

    <achievegoal name="citacoes_sejam_extraidas_da_pagina" exported="true">
        <parameter name="publicacao" class="Publicacao"/>
        <parameter name="pagina" class="String"/>
        <parameter name="citacoes" class="Integer" direction="out"/>
    </achievegoal>
</goals>

<plans>
<plan name="obter_as_citacoes_no_google_sholar">
    <parameter name="publicacoes" class="Publicacao[]">
        <goalmapping
ref="citacoes_das_publicacoes_sejam_recuperadas.publicacoes"/>
        </parameter>
        <parameter name="publicacoes_com_as_citacoes" class="Publicacao[]"
direction="out">
            <goalmapping
ref="citacoes_das_publicacoes_sejam_recuperadas.publicacoes_com_as_citacoes"/>
        </parameter>
        <body class="ObterAsCitacoesNoGoogleScholar"/>
        <trigger>
            <goal ref="citacoes_das_publicacoes_sejam_recuperadas"/>
        </trigger>
    </plan>

<plan name="recuperar_a_pagina_da_publicacao_no_google_scholar">

```

```

<parameter name="publicacao" class="Publicacao">
  <goalmapping ref="publicacao_seja_pesquisada.publicacao"/>
</parameter>
<parameter name="elo" class="String" direction="out">
  <goalmapping ref="publicacao_seja_pesquisada.elo"/>
</parameter>
<parameter name="pagina" class="String" direction="out">
  <goalmapping ref="publicacao_seja_pesquisada.pagina"/>
</parameter>
<body class="RecuperarAPaginaDaPublicacaoNoGoogleScholar"/>
<trigger>
  <goal ref="publicacao_seja_pesquisada"/>
</trigger>
</plan>

<plan name="parsear_a_pagina_da_publicacao">
  <parameter name="publicacao" class="Publicacao">
    <goalmapping ref="citacoes_sejam_extraidas_da_pagina.publicacao"/>
  </parameter>
  <parameter name="pagina" class="String">
    <goalmapping ref="citacoes_sejam_extraidas_da_pagina.pagina"/>
  </parameter>
  <parameter name="citacoes" class="Integer" direction="out">
    <goalmapping ref="citacoes_sejam_extraidas_da_pagina.citacoes"/>
  </parameter>
  <body class="ParsearAPaginaDaPublicacao"/>
  <trigger>
    <goal ref="citacoes_sejam_extraidas_da_pagina"/>
  </trigger>
</plan>
</plans>

...

</capability>

```

### C.3.4 Código-fonte da Classe ObterAsCitacoesNoGoogleScholar.java

Esta seção apresenta o código-fonte da classe ObterAsCitacoesNoGoogleScholar.java. Essa classe implementa o plano que coordena a obtenção das páginas das publicações no Google Scholar e a extração do número de citações. O parâmetro de entrada é uma lista de publicações e o parâmetro de saída é a lista de publicações com o número de citações incluído.

```
package lattesscholar.sma.scholar;

import jadex.runtime.IGoal;
import jadex.runtime.Plan;
import lattesscholar.sma.resources.Publicacao;

/**
 * @author Maurício Serrano
 */
public class ObterAsCitacoesNoGoogleScholar extends Plan {

    @Override
    public void body() {
        System.out.println("Scholar capability running ObterAsCitacoesNoGoogleScholar
plan.");

        Boolean offlineMode = (Boolean)
            this.getBeliefbase().getBelief("offline_mode").getFact();
        Publicacao[] publicacoes = (Publicacao[])
            this.getParameter("publicacoes").getValue();
        Publicacao[] publicacaoComAsCitacoes = new Publicacao[publicacoes.length];

        //Título: Obter as citacoes no Google Scholar
        //Ator: Google Scholar, Fornecedor de Citacoes
        //Objetivo: Citacoes das publicacoes sejam recuperadas
        //Recursos: Lista de publicacoes, página da publicacao no google scholar
        //Episódios:
```

```

// modo offline para testes

if (offlineMode) {

    for (int i = 0; i < publicacoes.length; i++) {
        Publicacao publicacao = publicacoes[i];

        int citacoes = (int) (java.lang.Math.random() * (publicacoes.length + 1));
        publicacao.setCitacoes(citacoes); //número de citações fictício
        publicacao.setElo("http://scholar.google.com");
        publicacaoComAsCitacoes[i] = publicacao;
    }
} else {

    // Episódio 01- Pesquisar cada uma das publicações no Google Scholar

    for (int i = 0; i < publicacoes.length; i++) {
        Publicacao publicacao = publicacoes[i];

        IGoal publicacaoSejaPesquisada =
            this.createGoal("publicacao_seja_pesquisada");
        publicacaoSejaPesquisada.getParameter("publicacao").setValue(publicacao);
        this.dispatchSubgoalAndWait(publicacaoSejaPesquisada);

        String elo = (String) publicacaoSejaPesquisada.getParameter("elo").getValue();
        String paginaDaPublicacao = (String)
            publicacaoSejaPesquisada.getParameter("pagina").getValue();

        // se não encontrou
        if (paginaDaPublicacao.equalsIgnoreCase("sem pagina")) {
            publicacao.setElo("sem elo");
            publicacao.setCitacoes(0);
            publicacaoComAsCitacoes[i] = publicacao;
        } else {

            // Episódio 02 - Encontrar na página o número de citações da publicação

```

```

publicacao.setElo(elo);

IGoal citacoesSejamExtraidasDaPagina =
    this.createGoal("citacoes_sejam_extraidas_da_pagina");

citacoesSejamExtraidasDaPagina.getParameter("publicacao").setValue(publicacao);
citacoesSejamExtraidasDaPagina.getParameter("pagina").setValue(paginaDaPublicacao);
    this.dispatchSubgoalAndWait(citacoesSejamExtraidasDaPagina);

    int citacoes = (Integer)
        citacoesSejamExtraidasDaPagina.getParameter("citacoes").getValue();

    publicacao.setCitacoes(citacoes);
    publicacaoComAsCitacoes[i] = publicacao;
    }
    }
}

// Episódio 03-Repassar as publicações com as citações para o
// agente Fornecedor de Citações

this.getParameter("publicacoes_com_as_citacoes").setValue(publicacaoComAsCitacoes);
}
}

```

### C.3.5 Código-fonte da Classe RecuperarAPaginaDaPublicacaoNoGoogle Scholar.java

Esta seção apresenta o código-fonte da página lattescholar.xhtml, que é a página inicial da aplicação *Web Lattes-Scholar*. Essa página representa o primeiro passo de ambos os processos de uso explicados no Apêndice A: o processo iniciado pelo consumidor com o nome do pesquisador e o processo iniciado pelo consumidor com a URL do currículo Lattes do pesquisador. A página lattescholar.xhtml está em português e a página análoga, step1.xhtml, em inglês, é apresentada na seção seguinte, Seção C.2.

```

package lattesscholar.sma.scholar;

import jadex.runtime.Plan;
import java.io.BufferedReader;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.URL;
import java.net.URLConnection;
import java.util.logging.Level;
import java.util.logging.Logger;
import lattesscholar.sma.resources.Publicacao;
import lattesscholar.sma.resources.StringFilter;

/**
 * @author Maurício Serrano
 */
public class RecuperarAPaginaDaPublicacaoNoGoogleScholar extends Plan {

    @Override
    public void body() {
        System.out.println("Scholar capability running
            RecuperarAPaginaDaPublicacaoNoGoogleScholar plan.");

        //Título: Recuperar a página da publicação no Google Scholar
        //Ator: Google Scholar, Fornecedor de Citações
        //Objetivo: Publicação seja pesquisada
        //Recursos: Publicação, URL de busca, página da publicação no Google Scholar
        //Episódios:

        // Episódio 01-Construir a URL de busca no Google Scholar

        Publicacao publicacao = (Publicacao) this.getParameter("publicacao").getValue();
        String paginaDaPublicacao = "";

        StringFilter stringFilter = new StringFilter();
    
```

```

String parte1 = "http://scholar.google.com/scholar?as_q=";
String titulo;
if (publicacao.getTitulo() == null) {
    titulo = publicacao.consultaString();
    titulo = stringFilter.removePunctuation(titulo);
    titulo = stringFilter.removeDoubleSpaces(titulo);
    titulo = stringFilter.replaceSpaceForPlus(titulo);
} else {
    titulo = stringFilter.replaceSpaceForPlus(publicacao.getTitulo());
}
String parte2 = "&num=20&btnG=Search+Scholar" +
    "&as_epq=&as_oq=&as_eq=&as_occt=any&as_sauthors=";
String autor = "";
String parte3 = "&as_publication=&as_ylo=";
String ano;
int valor = Integer.valueOf(publicacao.getAno());
if (valor < 1900 || valor > 2100) {
    ano = "";
} else {
    ano = publicacao.getAno();
}
String parte4 = "&as_yhi=&as_sdt=1." +
    "&as_sdtf=on&as_sdtf=&as_sdtf=5&hl=en";

String elo = parte1 + titulo + parte2 + autor + parte3 + ano + parte4;
URL urlDaBuscaNoGoogleScholar;

// Episódio 02-Fazer uma requisição HTTP para o Google Scholar com a
//      URL de busca

try {
    urlDaBuscaNoGoogleScholar = new URL(elo);

    URLConnection urlConn = urlDaBuscaNoGoogleScholar.openConnection();
    urlConn.addRequestProperty("User-Agent", "Mozilla/4.0 (compatible; MSIE
6.0; Windows NT 5.1)");

```



```

//String contentType = urlConn.getContentType();
//System.out.println("contentType:" + contentType);

InputStream inputstream = urlConn.getInputStream();
InputStreamReader inputStreamReader = new
    InputStreamReader(inputstream);
BufferedReader bufferedReader = new BufferedReader(inputStreamReader);
StringBuilder stringBuilder = new StringBuilder();

// Episódio 03-Receber a página da publicação no Google Scholar

String htmlLine;

while ((htmlLine = bufferedReader.readLine()) != null) {
    stringBuilder.append(htmlLine);
}

paginaDaPublicacao = stringBuilder.toString();

} catch (Exception ex) {

Logger.getLogger(RecuperarAPaginaDaPublicacaoNoGoogleScholar.class.getName())
    .log(Level.SEVERE, null, ex);
}

// Episódio 04-Repassar o elo e a página da publicação
this.getParameter("elo").setValue(elo);
this.getParameter("pagina").setValue(paginaDaPublicacao);
}
}

```

### C.3.6 Código-fonte da Página lattescholar.xhtml

Esta seção apresenta o código-fonte da página lattescholar.xhtml, que é a página inicial da aplicação *Web Lattes-Scholar*. Essa página representa o primeiro passo de ambos os processos de uso explicados no Apêndice A: o processo iniciado pelo consumidor com o nome do pesquisador e o processo iniciado pelo consumidor com a URL do currículo Lattes do pesquisador. A página lattescholar.xhtml está em português e a página análoga, step1.xhtml, em inglês, é apresentada na seção seguinte, Seção C.2.

```
package lattescholar.sma.scholar;

import jadex.runtime.Plan;
import java.util.regex.Matcher;
import java.util.regex.Pattern;
import lattescholar.sma.resources.Publicacao;
import lattescholar.sma.resources.StringFilter;

/**
 * @author Maurício Serrano
 */
public class ParsearAPaginaDaPublicacao extends Plan {

    @Override
    public void body() {
        System.out.println("Scholar capability running ParsearAPaginaDaPublicacao
plan.");

        //Título: Parsear a página da publicação
        //Ator: Google Scholar, Fornecedor de Citações
        //Objetivo: Citações sejam extraídas da página
        //Recursos: Publicações, página da publicacao no Google Scholar,
        //expressões regulares, patterns e matchers do Java
        //Episódios:
```

```

// Episódio 01-Encontrar na página os blocos com os textos das publicações
Publicacao publicacao = (Publicacao) this.getParameter("publicacao").getValue();
String pagina = (String) this.getParameter("pagina").getValue();

StringFilter stringFilter = new StringFilter();
String tituloPublicacao = publicacao.getTitulo();

if (tituloPublicacao == null || tituloPublicacao.equalsIgnoreCase("")) {
    this.getParameter("citacoes").setValue(new Integer(0));
} else {
    int citacoesInt = 0;

    String regexDiv = "<div class=gs_r>";
    String regexTitulo = ".*<a href=\".*\" onmousedown=\".*\">(.*?)</a></h3>.*";
    Pattern patternTitulo = Pattern.compile(regexTitulo);
    String regexCited = ".*>Cited by (\\d*)</a>.*";
    Pattern patternCited = Pattern.compile(regexCited);

    String[] split = pagina.split(regexDiv);
    String[] blocos = new String[split.length - 1];
    System.arraycopy(split, 1, blocos, 0, split.length - 1);
    blocos[blocos.length - 1] =
        blocos[blocos.length - 1].split("</a></div></div>")[0];

// Episódio 02-Para cada bloco de texto, procurar pelo título da publicação

for (int i = 0; i < blocos.length; i++) {
    System.out.println(blocos[i]);

    Matcher matcher = patternTitulo.matcher(blocos[i]);
    if (matcher.matches()) {
        System.out.println(matcher.group(1));
        String titulo = matcher.group(1);
        titulo = stringFilter.removeTags(titulo);
        titulo = titulo.replaceAll("[^a-zA-Z]", "");
        System.out.println(titulo);
    }
}

```

```
tituloPublicacao = stringFilter.removeTags(tituloPublicacao);
tituloPublicacao = tituloPublicacao.replaceAll("[^a-zA-Z]", "");

// Episódio 03-Se for a publicação pesquisada, acumular o número de citações
if (titulo.equalsIgnoreCase(tituloPublicacao)) {
    System.out.println("Iguar.");

    matcher = patternCited.matcher(blocos[i]);
    if (matcher.matches()) {
        String citacoesStr = matcher.group(1);
        citacoesInt = citacoesInt + Integer.valueOf(citacoesStr);
    }
} else {
    System.out.println("Não encontrou.");
}

// Repassar o número de citações da publicação pesquisada para o
// agente Fornecedor de Citações
System.out.println("Citações: " + citacoesInt);

Integer citacoes = new Integer(citacoesInt);
this.getParameter("citacoes").setValue(citacoes);
}
}
}
```