7
Referências Bibliográficas

[Aires00] Aires, R. V. X. Implementação, adaptação, combinação e avaliação de etiquetadores para o português do Brasil. Dissertação de Mestrado, ICMC-USP, São Carlos, SP. 2000.

[Audy04] Audy, Jorge; Evaristo, Roberto; Watson-Manheim, Mary. "Distributed Analysis: the Last Frontier?" In: 37th Hawaii International Conference on System Sciences, 2004. Proceedings. pgs. 1-9.

[Audy04a] Audy, Jorge L. N. & Lopes, Leandro. "Towards a reference model for requirement engineering in distributed software development". In: CAiSE - 16th International Conference on Advanced Information Systems Engineering, Riga, Letonia. 2004. Proceedings.

[Baeza-Yates99] Baeza-Yates, R. & Ribeiro Neto, B. "Modern Information Retrieval". New York: ACM Press, 1999.

[Baniassad04] Baniassad, Elisa & Clarke, Sióbhan. "Theme: an Approach for Aspect-Oriented Analysis and Design". In: the International Conference on Software Engineering, 2004. Proceedings.

[Bardin77] Bardin, L. Análise de conteúdo. Lisboa, Ed. 70, 1977. 225 p.

[Bellifemine01] Bellifemine, F.; Poggi, A.; Rimassa, G. JADE: a FIPA2000 compliant agent development environment. In: Fifth international Conference on Autonomous Agents (AGENTS '01). Proceedings. ACM Press, New York, NY, 2001. pp. 216-217.

[Bianchi02] Bianchi, A.; Caivano, D.; Lanubile, F.; Rago, F. & Visaggio, G. "An Empirical Study of Distributed Software Maintenance". In: International Conference on Software Maintenance (ICSM.02). Proceedings.

[Blackburn01] BLACKBURN, M. R.; BUSSER, R.; NAUMAN, A. Removing Requirement Defects and Automating Test. Software Productivity Consortium NFP, 2001. Disponível em <http://www.software.org/ pub/taf/downloads/ RemovingRequirementDefects.pdf> Acesso em 26 nov 2002.

[Boehm76] Boehm, Barry. Software Engineering, IEEE Transactions on Computers, v. C-25, Dec 1976.

[Bohem01] Bohem, B. & Basili, V. Software Defect Reduction Top 10 List. In: IEEE Computer, vol. 34, nº 1, jan. 2001. pp. 135-137.

[Booch00] Booch, G. ; Rumbaugh, J. & Jacobson, I. "UML: guia do usuário". Rio de Janeiro: Campus, 2000. 472 p. ISBN 8535205624

[Boyd05] Boyd, S.; Zowghi, D. & Farroukh, A. "Measuring the expressiveness of a Constrained Natural Language: an Empirical Study". In: 13th IEEE

International Conference on Requirements Engineering (RE´05). Proceedings.

[Caldas01] Caldas Jr, J.; Imamura, C.Y.M.; Rezende, S.O. Avaliação de um Algoritmo de Stemming para o Língua Portuguesa. In: 2nd Congress of Logic Applied to Technology, 2001. Proceedings. Vol. 2. pp. 267–274.

[Carmel99] Carmel, E. "Global Software Teams". Prentice-Hall, 1999.

[Chang01] Chang, C.; Cai, L. "Agent based Requirements Evolution over the Internet". In: IEEE Workshop on Software Engineering on the Internet, The IEEE-CS/IPSJ 2001 Symposium on Applications and the Internet (SAINT 2001), Jan. 8-12, 2001. pp. 83-88.

[Chaves03] Chaves, M. S. Um estudo e apreciação sobre algoritmos de stemming para a língua portuguesa. In: IX Jornadas Iberoamericanas de Informática. Cartagena de Indias - Colômbia, 11-15 agosto de 2003.

[Chen05] Chen, K.; Zhang, W.; Zhao, H. & Mei, Hong. An Approach to Constructing Feature Models Based on Requirements Clustering. In: 13th IEEE International Conference on Requirements Engineering (RE´05). Proceedings. pp. 31-40.

[Cherry04] Cherry, S. & Robillard, P. "Communication Problems in Global Software Development: Spotlight on a New Field of Investigation". In: Third International Workshop on Global Software, May 24, 2004, Edinburgh, Scotland. Proceedings. http://gsd2004.uvic.ca/

[Cysneiros03] Cysneiros, Luiz Márcio; Yu, Eric. Requirements Engineering for Large-Scale Multi-Agent Systems. In: 2nd International Workshop on Software Engineering for Large-Scale Multi-Agent Systems, Portland, Oregon - USA, 2003. Proceedings. pp. 36-57.

[Dag01] Dag, J. N.; Regnell, B.; Carlshamre, P.; Andersson, M. & Karlsson, J. "Evaluating Automated Support for Requirements Similarity Analysis in Market-Driven Development". In: 7th Int. Workshop on Requirements Engineering: Foundation for Software Quality, June 4-5 2001, Interlaken, Switzerland. Proceedings.

[Daile96] Daille, B. "Study and Implementation of Combined Techniques for Automatic Extraction of Terminology". In: Klavans, J., Resnik, P. The Balancing ACT- Combining Symbolic and Statistical Approaches to Language, The MIT Press, 1996. pp. 49-66.

[Damian03] Damian, D.; Eberlein, A.; Shaw, M. & Gaines, B. "Facilitation in Distributed Requirements Engineering". Requirements Engineering Journal, 8(1), 2003, pags 23-41.

[Damian03a] Damian, D. & Zowghi, D. "RE challenges in multi-site software development organizations". Requirements Engineering Journal, 8(3),2003, pgs. 149-160.

[Davis93] DAVIS, Alan et al. Identifying and Measuring Quality in Software Requirements Specifications**.** In: IEEE-CS INTERNATIONAL SOFTWARE METRICS SYMPOSIUM, 1., 1993, Baltimore. **Proceedings**. Los Alamitos, California: IEEE Computer Society Press, may 1993, p. 141-152.

[Dellen96] Delle, B. & Maurer, F. "Integrating Planning and Execution in

Software Development Process". In: Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE'96). Proceedings, pp. 170–176.

[Dhavachelvan04] Dhavachelvan, P.; Uma, G.V. "Reliability Enhancement in Software Testing- An Agent-Based Approach for Complex Systems". Lecture Notes in Computer Science, vol. 3356,jan 2004. pp. 282 - 291.

[El-Emam00] El-Emam, K.& Birk, A. Validating the ISO/IEC 15504 Measure of Software Requirements Analysis Process Capability. Journal of Systems and Software, 51(2). pp. 119-149.

[Fagan86] FAGAN, M. E. Advances in Software Inspections. **IEEE Transactions on Software Engineering**, vol. SE-12, nº 7, p. 744-751, july 1986.

[Faltings00] Faltings, B. Intelligente Agents: Software Technology for the new Millennium. In: Informatik 1/2000, Editorial. pags. 2-5. Disponível em <http://www.svifsi.ch/revue/pages/issues/n001/no001.html>. Acesso em 11.12.2003.

[Fantechi05] Fantechi, A.; Spinicci, E. **A Content Analysis Technique for Inconsistency Detection in Software Requirements Documents**. Anais do WER05 - Workshop em Engenharia de Requisitos, Porto, Portugal, Junho 13-14, 2005. pp 245-256.

[Ferber99] Ferber, Jacques. Multi-Agent Systems: an Introduction to Distributed Artificial Intelligence. Pearson Edicational Limited, 1999.

[Gaeta02] Gaeta, M. & Ritrovato, P. "Generalised Environment for Process Management in Cooperative Software Engineering". In: 26th Annual International Computer Software and Applications Conference (COMPSAC'02). Proceedings.

[Gershenson99] Gershenson, J. A. & Stauffer, L. A. "A taxonomy for design requirements from corporate customers". Research in Engineering Design, vol 11. pp 103-115.

[Gervasi02] Gervasi, V. & Nuseibeh,B. Lightweight validation of natural language requirements". In: Software Practice and Experience, vol. 32, no. 2, 2002. pp. 113-133

[Gonzalez05] Gonzalez, M.A.I. "Termos e Relacionamentos em Evidência na Recuperação de Informação". Tese de doutorado, Programa de Pós-Graduação em Computação, Universidade Federal do Rio Grande do Sul, Porto Alegre, RS. 2005.

[Gorton96] Gorton, I. & Motwani, S. "Issues in Co-operative Software Engineering Using Globally Distributed Teams". In: Information and Software Technology Journal ,vol 38(10), 1996. págs. 647-655.

[Gruenbacher01] Gruenbacher, P.; Egyed, A. & Medvidovic, N. "Dimensions of Concerns in Requirements Negotiation and Architecture Modeling". In: International Conference on Software Engineering - ICSE 2001. Proceedings.

[Grundy05] Grundy, J.; Ding, G. & Hosking, J. "Deployed software component testing using dynamic validation agents". The Journal of Systems and

Software 74 (2005). pags 5-14.

[Haendchen05] Haendchen Filho, A. A Middleware Framework for Buiding Multi-Agent Systems in the Internet. Ph.D. Thesis, Pontifícia Universidade Católica do Rio de Janeiro PUC-Rio, Rio de Janeiro, RJ, Brasil, 2005.

[Haendchen07] Haendchen F., A.; Prado, H. A.; Lucena, C. J. P. A WSA-based architecture for building multi-agent systems. In: International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2007). Honolulu, Hawai'i. Proceedings. a ser publicado.

[Hammer96] HAMMER, T. et al. Requirements Metrics for Risk Identification. In: SOFTWARE ENGINEERING WORKSHOP, 21st, 1996, NASA-GSFC, MD. **Proceedings**. Disponível em <http://satc.gsfc.nasa.gov/support/SEW_DEC96/ sel.PDF>. Acesso em 21 out 2002.

[Hardy04] Hardy, C.; Harley, B. & Phillips, N. "Discourse Analysis and Content Analysis: Two Solitudes?". In: Qualitative Methods, vol. 2:1, Spring 2004. pp.19-22.

[Harmain00] Harmain, H. M. & Gaizauskas, R. "CM-Builder: An Automated NL-based CASE Tool". In: 15th IEEE International Conference on Automated Software Engineering (ASE'2000), 2000. Proceedings. pp. 45-53.

[Herbsleb03] Herbsleb, J. & Mockus, A. "An empirical study of speed and communication in globally distributed software development". IEEE Transactions on Software Engineering, vol 29(6), jun 2003. pgs. 481-494.

[Himmelspach03] Himmelspach, J.; Rohl, M. & Uhrmacher, A.M. "Simulation for testing software agents - an exploration based on James". In: Simulation Conference, 2003. Proceedings. pp. 799-807.

[Hoskinson05] Hoskinson, A. "Creating the Ultimate Research Assistant". IT Systems Perspective, IEEE Computer Society, november 05. pp. 97-99.

[Hsia92] Hsia, P. & Gupta, A. "Incremental Delivery Using Abstract Data Types and Requirements Clustering". In: Second International Conference on Systems Integration. Proceedings. IEEE Computer Society, June 1992. pp. 137-150.

[IEEE98] Institute of Electrical and Electronics Engineers. "System Requirements Specifications". Document Number: IEEE 830-1998, 1998.

[Jennings00] Jennings, N.R.. "On agent-based software engineering". In: Artificial Intelligence, vol. 117 (2000). pp. 277–296.

[Jennings01] Jennings, N.R., and Wooldridge, M.J.: Agent-Oriented Software Engineering. In: Bradshaw, J. (ed.): Handbook of Agent Technology. AAI/MIT Press (2001). Disponível em <http://www.ecs.soton.ac.uk/~nrj/download-files/agt-handbook.pdf>. Acesso em 12.01.04.

[Jones96] JONES, T. Capers. Applied Software Measurement: Assuring Productivity and Quality. New York, McGraw-Hill, 1996.

[Kotonya98] Kotonya, G. e Sommerville, I. Requirements Engineering: process and techniques. John Willey & Sons Ltd, 1998.

[Kowalski97] Kowalski, G. Information retrieval systems : theory and implementation. Boston : Kluwer Academic, 1997. 282 p.

[Leffingwell97] Leffingweel, D. Calculating the return on investment from more effective requirements management. American Programmer, vol. 10, nº5, abr 1997. pp.13-16.

[Leite90] - Leite, J.C.S.P.; Franco, A. P. "O uso de hipertexto na elicitação de linguagens de da aplicação". In: 4º Simpósio Brasileiro de Engenharia de Software, 1990. Anais. pp.124-133.

[Leite91] Leite, J.C.S.P.; Freeman, P.A. "Requirements Validation through Viewpoint Resolution". In: IEEE Transactions on Software Engineering: vol. 17, nº 12. pp. 1253-1269.

[Leite93] Leite, J.C.S.P. & Franco, A.P.M. "A Strategy for Conceptual Model Acquisition". In: First IEEE International Symposium on Requirements Engineering, San Diego, Ca, IEEE Computer Society Press, 1993. Proceedings. pp. 243-246.

[Leite94] Leite, J. C. S. P. Engenharia de Requisitos. PUC-Rio, Rio de Janeiro, 1994. Notas de aula.

[Leite95] Leite, J.C.S.P. & Oliveira, A.P. A Client Oriented Requirements Baseline – In: Second IEEE International Symposium on Requirements Engineering (RE'95), 1995. Proceedings. Los Alamitos: IEEE Computer Society Press, 1995. pp.108-115.

[Leite01] Leite, J. C. S. P. In: "Qualidade e Produtividade em Software", eds. Kival Chaves Weber et al. São Paulo, Ed. Makron Books, 2001.

[Li03] Li, Y.; Shen, W. & Ghenniwa, H. Collaborative and Proactive Data Agent for Distributed Design Environments. In: Journal of Integrated Design and Process Science, june 2003, Vol. 7, No. 2, pp. 71-85

[Li03a] Li, Y.; Shen, W. & Ghenniwa, H. Integrated description for Web service-oriented agents in e-Marketplaces. In: BASeWEB 2003 Workshop. Proceedings. Disponível em http://www.cs.unb.ca/baseweb/baseweb03/papers/li_ghenniwa_shen_revision-may12.pdf. Acesso em 12.06.2005.

[Liberato97] Liberato, Y. G. "A estrutura do SN em português: uma abordagem cognitiva". Tese de doutorado, 1997. UFMG, Departamento de Lingüística, Belo Horizonte.

[Lopes05] Lopes, L.; Prikladnicki, R.; Audy, J. & Majdenbaum, A. "Requirements Specification in Distributed Software Development - A Process Proposal". In: 38th Hawaii International Conference on System Sciences - HICSS. Hawaii, USA, 2005. Proceedings.

[Lowe04] Lowe, W. "Content analysis and its place in the (Methodological) scheme of things". In: Qualitative Methods, vol. 2:1, Spring 2004. pp.25-27.

[Luhn58] Luhn, H.P. The automatic creation of literature abstracts, IBM Journal of Research and Development, vol. 2, 1958. pp. 159-165.

[Manning99] Manning, Christopher D.; Schütze, Heinrich. Foundations of statistical natural language processing. Cambridge: MIT Press, c1999. 680

p. ISBN 0262133601 (enc.)

[Mason97] Mason, O. & Tufis, D. "Probabilistic Tagging in a Multi-lingual Environment: Making an English Tagger Understand Romanian". In: Third European TELRI Seminar, Montecatini, Italy, 1997. Proceedings.

[Matsubara03] Matsubara, Edson T.; Martins, Claudia A. & Monard, Maria C. PreTexT: uma ferramenta para pré-processamento de textos utilizando a abordagem bag-of-words. Relatório técnico nº 209, série Relatórios Técnicos do ICMC, São Carlos. 2003.

[Meadow00] Meadow, C. T.; Boyce, Bert & Kraft, Donald H. Text information retrieval systems. 2 ed. San Diego : Academic Press, 2000. 364 p.

[Message01] MESSAGE: Methodology for Engineering Systems of Software Agents - Final guidelines for the relevant problems areas where agent technology is appropriate. EURESCOM Participants in Project P907-GI (2001).

[Moraes99] Moraes, R. Análise de Conteúdo. In: Revista Educação, Porto Alegre, vol. 22 nº 37, março 1999. pp. 7-32.

[Orengo01] Orengo. V & Huyck, C. A stemming algorithm for the Portuguese language. In: Proceedings of the 8th International Symposium on String Processing and Information Retrieva l (SPIRE) 2001. pp 186–193.

[Overainder04] Overeinder, B. J.; Brazier, F. Scalable Middleware Environment for Agent-Based Internet Applications. Data Knowledge Engineering, 41(2-3):229-245, 2004. Elsevier Science Publishers, Amsterdam, The Netherlands, 2004.

[Ozkaya06] Ozkaya, I. Representing Requirements Relationships. In: Fisrt International Workshop on Requirements Engineering Visualization (REV´06). Proceedings.

[Palmer92] Palmer, J.D. & Liang, Y. "Indexing and Clustering of Software Requirements Specifications". In: Information and Decision Technologies vol. 18, nº 4. pp 283-299.

[Paré99] Paré, G. & Dubé, L. “Virtual Teams: An Exploratory Study of Key Challenges and Strategies”. In: 20th International Conference on Information Systems, dez 1999, Charlotte, North Carolina, United States. Proceedings. pp. 479-483.

[Park00] Park, S.; Kim, H.; Ko, Y. & Seo, J. Implementation of an efficient requirements-analysis supporting system using similarity measure techniques. In: Information and Software Technology 42 (2000). pp. 429–438.

[Parreiras03] Parreiras, F. "O uso de sintagmas nominais como fonte de descritores para textos de periódicos científicos". Escola de Ciência da Informação. Belo Horizonte, 2003. Disponível em http://www.fernando.parreiras.nom.br/publicacoes/sn.pdf.

[Pepper00] Pepper, S. "The TAO of Topic Maps, finding the way in the age of infoglut". In: XML Europe Conference, Paris, 2000. Proceedings.

[Pepper01] Pepper, Steve & Moore, Graham. "XML Topic Maps (XTM) 1.0". http://www.topicmaps.org/xtm/1.0/, Mar 2001. TopicMaps.Org Specification.

[Pérez03] Pérez, C. C. C.; Gasperin, C. & Vieira, R. "Extração semi-automática de conhecimento a partir de textos". In: IV Encontro Nacional de Inteligência Artificial (ENIA 2003), Campinas, 2003. Anasi da SBC, 2003, v. 7, pp.193-202.

[Perini98] Perini, M. A. "Gramática descritiva do português". 3ªed. - São Paulo: Ática, 1998. 380p. ISBN 8508055501

[Ponnurangam05] Ponnurangam, D. Uma, G. V. Fuzzy complexity assessment model for resource negotiation and allocation in agent-based software testing framework
Expert Systems with Applications, vol. 29,no 1, July 2005. pp. 105-11.

[Porter80] Porter. M. F. An algorithm for suffix stripping. Program, 14(3) pp 130−137. Disponível em http://www.tartarus.org/~martin/PorterStemmer/. Acesso em 06.04.2004.

[Porter95] PORTER, A. A. ; VOTTA JR, L. G.; BASILI, V. Comparing Detection Methods for Software Requirements Inspections: a replicated experiment. **IEEE Transactions on Software Engineering**, vol. 21, nº 6, p. 563-575, june 1995.

[Prikladnicki03] Prikladnicki, R., Audy, J.L.N., & Evaristo, R. "Global software development in practice: lessons learned". Software Process: Improvement and Practice, 8(4), 2003. pp. 267-281.

[Prikladnicki04] Prikladnicki, R. & Audy, J. "MuNDDoS - Um Modelo de Referência para Desenvolvimento Distribuído de Software". In: XVIII Simpósio Brasileiro de Engenharia de Software - 2004 - Brasília, DF, Brasil. Anais. pgs. 289-304.

[Rosenberg98] ROSENBERG, Linda. et al. Requirements, Testing, and Metrics. In: PACIFIC NORTHWEST SOFTWARE QUALITY CONFERENCE, 15th, 1998, Utah. Proceedings. Disponível em <http://satc.gsfc.nasa.gov/support/ PNSCQ _OCT98/requirements_testing_and_metrics.html>. Acesso em 12 out 2002.

[Ross77] Ross, Douglas & Schoman, A. "Structured analysis for requirements definition". IEEE Transactions on Software Engineering. 1977. Vol. 3(1), pp. 6–15.

[Salton83] Salton, Gerard. Introduction to modern information retrieval. New York, NY : McGraw-Hill, c1983. 448 p.

[Salton88] Salton, G. & Buckley, C. Term-weighting approaches in automatic text retrieval. Information Processing and Management, vol. 24 (5), 1988. pp. 513–523.

[Sampaio05] Sampaio, A.; Chitchyan, R.; Rashid, A. & Rayson, P. "EA-Miner: a Tool for Automating Aspect-Oriented Requirements Identification". In: ASE 2005.

[Sardinha04] Sardinha, A. P. B. "Lingüística de Corpus". São Paulo: Ed. Manole. ISBN: 1676-4. 2004. 410 páginas.

[Sayão05] Sayão, M. & Leite, J. C. S. P. "Uso de Agentes no Processo de Requisitos em Ambientes Distribuídos de Desenvolvimento". In: Workshop de Engenharia de Requisitos, Lisboa, Portugal, 2005. Anais.

[Shull00] SHULL, F.; RUS, I.; BASILI, V. How Perspective-Based Reading can Improve Requirements Inspections. **IEEE Computer**, vol. 33, nº 7, p. 73-79, july 2000.

[Silva06] Silva, Lyrene F. Uma estratégia orientada a aspectos para modelagem de requisitos. 2006. 222 f.Tese (Doutorado em Informática)-Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, RJ.

[Sommerville04] Sommerville, I. Software Engineering. Pearson Educational Limited, seventh edition, 2004.

[Sommerville98] Sommerville, I.; Sawyer, P. & Viller, S. "Viewpoints for requirements elicitation: a practical approach". In: Third International Conference on Requirements Engineering, 1998. Proceedings. pp. 74 - 81

[Standish04] Standish Group. The CHAOS Report 2004.

[SWEBOK04] Guide to the Software Engineering Body of Knowledge. 2004.

[Teline03] Teline, M. F.; Almeida, G. M. B. & Aluísio, S. M. "Extração Manual e Automática de Terminologia: Comparando Abordagens e Critérios". In: 16th Brazilian Symposium on Computer Graphics and Image Processing - SIBGRAPI 2003. Proceedings.

[Terra04] Terra, E. "Lexical Affinities and Language Applications". PhD Thesis, University of Waterloo, Canada. 2004.

[Thrane80] THRANE, Torben. Referential-Semantic Analysis: Aspects of a Theory of Linguistic Reference. Londres: Cambridge University Press, 1980. 256p.

[Tufis98] Tufis, D. & Mason, O. Tagging Romanian Texts: a Case Study for QTAG, a Language Independent Probabilistic Tagger. In: First International Conference on Language Resources & Evaluation, Granada, Spain. Proceedings. pp. 589-596.

[Vieira01] Vieira, R. & Lima, V. L. S. (2001). "Lingüística Computacional: Princípios e Aplicações". In: As Tecnologias da Informação e a questão social: anais. Carlos Eduardo Ferreira (Ed.) Fortaleza, SBC. ISBN 85-88442-03-5 (v.2). pp 47-88.

[Wilson97] WILSON, W.M.; ROSENBERG, L.H.; HYATT, L. E. Automated Analysis of Requirement Specifications. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING (IASTED), 1997, Boston, MA. Proceedings. Disponível em <http://satc.gsfc.nasa.gov/support/ICSE_MAY97/ arm/ICSE97-arm.htm>. Acesso em 13 set 2002.

[Witten00] Witten, I. H. & Frank, E. Data mining : practical machine learning tools and techniques with java implementations. San Francisco, CA : Morgan Kaufmann, 2000. 371 p.

[Wives04] Wives, Leandro. "Utilizando conceitos como descritores de textos para o processo de identificação de conglomerados (clustering) de documentos". Tese de doutorado, UFRGS. 2004.

[Wongthongtham06] Wongthongtham, P.; Chang, E.; Dillon, T. S. & Sommerville, I., Ontology-based multi-site software development. Journal of Systems Architecture (2006), doi:10.1016/j.sysarc.2006.06.008

[Wooldridge99a] Wooldridge, M. Intelligent Agents. In: Weiss, Gerhard (Ed.). Multiagent Systems - A Modern Approach. MIT Press, 1999. pp. 27-77.

[Wu06] Wu, S.; Ghenniwa, H.; Zhang, Y & Shen, W. Personal assistant agents for collaborative design environments. In: Computers in Industry (2006), doi:10.1016/j.compind.2006.04.010

[Yu95] Yu, E. Modelling Strategics Relationships for Process Reengineering. PhD Thesis, University of Toronto, 1995.

[Yu02] Yu, E., Agent-Oriented Modelling: Software Versus the World. In: Agent-Oriented Software Engineering AOSE-2001 Workshop Proceedings, Montreal, Canada - May 29th 2001. LNCS 2222.

[Zipf49] Zipf, G. Human Behaviour and the Principle of Least Effort. Addison-Wesley, 1949.

[Zowghi02] Zowghi, Didar. "Does Global Software Development need a different Requirements Engineering Process?" In: International Workshop on Global Software Development – ICSE 2002, Orlando, Florida, USA, 2002. Proceedings. pgs. 53-55.

Apêndice A
Padrões para extração de sujeitos/atores: doc. Escalas

<w POS="N">*ente <w POS="PRP">* <w POS="N">*
<w POS="N">*entes <w POS="PRP">* <w POS="N">*
<w POS="N">*enta <w POS="PRP">* <w POS="N">*
<w POS="N">*entas <w POS="PRP">* <w POS="N">*
<w POS="N">*or <w POS="PRP">* <w POS="N">*
<w POS="N">*ores <w POS="PRP">* <w POS="N">*
<w POS="N">*ora <w POS="PRP">* <w POS="N">*
<w POS="N">*oras <w POS="PRP">* <w POS="N">*
<w POS="N">*eiro <w POS="PRP">* <w POS="N">*
<w POS="N">*eiros <w POS="PRP">* <w POS="N">*
<w POS="N">*eira <w POS="PRP">* <w POS="N">*
<w POS="N">*eiras <w POS="PRP">* <w POS="N">*
<w POS="N">*ário <w POS="PRP">* <w POS="N">*
<w POS="N">*ários <w POS="PRP">* <w POS="N">*
<w POS="N">*ária <w POS="PRP">* <w POS="N">*
<w POS="N">*árias <w POS="PRP">* <w POS="N">*
<w POS="N">*es <w POS="PRP">* <w POS="N">*
<w POS="N">*eses <w POS="PRP">* <w POS="N">*
<w POS="N">*esa <w POS="PRP">* <w POS="N">*
<w POS="N">*esas <w POS="PRP">* <w POS="N">*
<w POS="N">*eus <w POS="PRP">* <w POS="N">*
<w POS="N">*euses <w POS="PRP">* <w POS="N">*
<w POS="N">*eusa <w POS="PRP">* <w POS="N">*
<w POS="N">*eusas <w POS="PRP">* <w POS="N">*
<w POS="N">*ia <w POS="PRP">* <w POS="N">*
<w POS="N">*ias <w POS="PRP">* <w POS="N">*
<w POS="N">*ente <w POS="N">*
<w POS="N">*entes <w POS="N">*
<w POS="N">*enta <w POS="N">*
<w POS="N">*entas <w POS="N">*
<w POS="N">*or <w POS="N">*
<w POS="N">*ores <w POS="N">*
<w POS="N">*ora <w POS="N">*
<w POS="N">*oras <w POS="N">*
<w POS="N">*eiro <w POS="N">*
<w POS="N">*eiros <w POS="N">*
<w POS="N">*eira <w POS="N">*
<w POS="N">*eiras <w POS="N">*
<w POS="N">*ário <w POS="N">*
<w POS="N">*ários <w POS="N">*
<w POS="N">*ária <w POS="N">*
<w POS="N">*árias <w POS="N">*
<w POS="N">*ês <w POS="N">*
<w POS="N">*êses <w POS="N">*
<w POS="N">*eses <w POS="N">*
<w POS="N">*esa <w POS="N">*
<w POS="N">*esas <w POS="N">*
<w POS="N">*eus <w POS="N">*
<w POS="N">*euses <w POS="N">*
<w POS="N">*eusa <w POS="N">*

<w POS="N">*eusas <w POS="N">*
<w POS="N">*ia <w POS="N">*
<w POS="N">*ias <w POS="N">*
<w POS="N">*ente
<w POS="N">*entes
<w POS="N">*enta
<w POS="N">*entas
<w POS="N">*or
<w POS="N">*ores
<w POS="N">*ora
<w POS="N">*oras
<w POS="N">*eiro
<w POS="N">*eiros
<w POS="N">*eira
<w POS="N">*eiras
<w POS="N">*ário
<w POS="N">*ários
<w POS="N">*ária
<w POS="N">*árias
<w POS="N">*ês
<w POS="N">*êses
<w POS="N">*eses
<w POS="N">*esa
<w POS="N">*esas
<w POS="N">*eus
<w POS="N">*euses
<w POS="N">*eusa
<w POS="N">*eusas
<w POS="N">*ia
<w POS="N">ias

Apêndice B
Padrões para extração de sujeitos/atores: doc. SELIC

<w POS="N">*ente <w POS="PRP">* <w POS="N">*
<w POS="N">*entes <w POS="PRP">* <w POS="N">*
<w POS="N">*enta <w POS="PRP">* <w POS="N">*
<w POS="N">*entas <w POS="PRP">* <w POS="N">*
<w POS="N">*or <w POS="PRP">* <w POS="N">*
<w POS="N">*ores <w POS="PRP">* <w POS="N">*
<w POS="N">*ora <w POS="PRP">* <w POS="N">*
<w POS="N">*oras <w POS="PRP">* <w POS="N">*
<w POS="N">*eiro <w POS="PRP">* <w POS="N">*
<w POS="N">*eiros <w POS="PRP">* <w POS="N">*
<w POS="N">*eira <w POS="PRP">* <w POS="N">*
<w POS="N">*eiras <w POS="PRP">* <w POS="N">*
<w POS="N">*ário <w POS="PRP">* <w POS="N">*
<w POS="N">*ários <w POS="PRP">* <w POS="N">*
<w POS="N">*ária <w POS="PRP">* <w POS="N">*
<w POS="N">*árias <w POS="PRP">* <w POS="N">*
<w POS="N">*ês <w POS="PRP">* <w POS="N">*
<w POS="N">*eses <w POS="PRP">* <w POS="N">*
<w POS="N">*esa <w POS="PRP">* <w POS="N">*
<w POS="N">*esas <w POS="PRP">* <w POS="N">*
<w POS="N">*eus <w POS="PRP">* <w POS="N">*
<w POS="N">*euses <w POS="PRP">* <w POS="N">*
<w POS="N">*eusa <w POS="PRP">* <w POS="N">*
<w POS="N">*eusas <w POS="PRP">* <w POS="N">*
<w POS="N">*ia <w POS="PRP">* <w POS="N">*
<w POS="N">*ias <w POS="PRP">* <w POS="N">*
<w POS="N">*ente <w POS="CPR">* <w POS="N">*
<w POS="N">*entes <w POS="CPR">* <w POS="N">*
<w POS="N">*enta <w POS="CPR">* <w POS="N">*
<w POS="N">*entas <w POS="CPR">* <w POS="N">*
<w POS="N">*or <w POS="CPR">* <w POS="N">*
<w POS="N">*ores <w POS="CPR">* <w POS="N">*
<w POS="N">*ora <w POS="CPR">* <w POS="N">*
<w POS="N">*oras <w POS="CPR">* <w POS="N">*
<w POS="N">*eiro <w POS="CPR">* <w POS="N">*
<w POS="N">*eiros <w POS="CPR">* <w POS="N">*
<w POS="N">*eira <w POS="CPR">* <w POS="N">*
<w POS="N">*eiras <w POS="CPR">* <w POS="N">*
<w POS="N">*ário <w POS="CPR">* <w POS="N">*
<w POS="N">*ários <w POS="CPR">* <w POS="N">*
<w POS="N">*ária <w POS="CPR">* <w POS="N">*
<w POS="N">*árias <w POS="CPR">* <w POS="N">*
<w POS="N">*ês <w POS="CPR">* <w POS="N">*
<w POS="N">*eses <w POS="CPR">* <w POS="N">*
<w POS="N">*esa <w POS="CPR">* <w POS="N">*
<w POS="N">*esas <w POS="CPR">* <w POS="N">*
<w POS="N">*eus <w POS="CPR">* <w POS="N">*
<w POS="N">*euses <w POS="CPR">* <w POS="N">*
<w POS="N">*eusa <w POS="CPR">* <w POS="N">*
<w POS="N">*eusas <w POS="CPR">* <w POS="N">*

<w POS="N">*ia <w POS="CPR">* <w POS="N">*
<w POS="N">*ias <w POS="PRP">* <w POS="N">*
<w POS="N">*ente <w POS="N">*
<w POS="N">*entes <w POS="N">*
<w POS="N">*enta <w POS="N">*
<w POS="N">*entas <w POS="N">*
<w POS="N">*or <w POS="N">*
<w POS="N">*ores <w POS="N">*
<w POS="N">*ora <w POS="N">*
<w POS="N">*oras <w POS="N">*
<w POS="N">*eiro <w POS="N">*
<w POS="N">*eiros <w POS="N">*
<w POS="N">*eira <w POS="N">*
<w POS="N">*eiras <w POS="N">*
<w POS="N">*ário <w POS="N">*
<w POS="N">*ários <w POS="N">*
<w POS="N">*ária <w POS="N">*
<w POS="N">*árias <w POS="N">*
<w POS="N">*ês <w POS="N">*
<w POS="N">*êses <w POS="N">*
<w POS="N">*eses <w POS="N">*
<w POS="N">*esa <w POS="N">*
<w POS="N">*esas <w POS="N">*
<w POS="N">*eus <w POS="N">*
<w POS="N">*euses <w POS="N">*
<w POS="N">*eusa <w POS="N">*
<w POS="N">*eusas <w POS="N">*
<w POS="N">*ia <w POS="N">*
<w POS="N">*ias <w POS="N">*
<w POS="N">*ão <w POS="N">*
<w POS="N">*ões <w POS="N">*
<w POS="N">*ente
<w POS="N">*entes
<w POS="N">*enta
<w POS="N">*entas
<w POS="N">*or
<w POS="N">*ores
<w POS="N">*ora
<w POS="N">*oras
<w POS="N">*eiro
<w POS="N">*eiros
<w POS="N">*eira
<w POS="N">*eiras
<w POS="N">*ário
<w POS="N">*ários
<w POS="N">*ária
<w POS="N">*árias
<w POS="N">*ês
<w POS="N">*êses
<w POS="N">*eses
<w POS="N">*esa
<w POS="N">*esas
<w POS="N">*eus
<w POS="N">*euses
<w POS="N">*eusa
<w POS="N">*eusas
<w POS="N">*ia
<w POS="N">*ias
<w POS="N">* <w POS="ADJ">* <w POS="ADV">*
<w POS="N">* <w POS="ADJ">*

Apêndice C
Dicionário de Requisitos Não-Funcionais

```
<?xml version="1.0" encoding="UTF-8"?>
<dictionary style="050805" patternengine="substring">
<cnode name="RNFS" desc="dicionário de requisitos não funcionais">
<cnode name="requisitos_nao_funcionais" desc="">
<cnode name="externos" desc="relacionados a aspectos externos ao
sistema">
<cnode name="comunicação sistemas externos" desc="outros sistemas
com os quais existirá troca de informações">
<pnode name="reply"/>
<pnode name="request"/>
</cnode>
<cnode name="custos" desc="restrições orçamentárias para o
sistema">
</cnode>
<cnode name="integração sistemas externos" desc="sistemas já
existentes com os quais haverá integração">
<pnode name="ace"/>
<pnode name="galileo"/>
<pnode name="sap"/>
</cnode>
<cnode name="interoperabilidade" desc="outros sistemas com os
quais haverá interação">
<pnode name="exportar"/>
<pnode name="exportação"/>
<pnode name="exportação de dados"/>
<pnode name="importar"/>
<pnode name="importar dados"/>
<pnode name="importação"/>
<pnode name="importação de dados"/>
<pnode name="integração"/>
<pnode name="migração de dados"/>
<pnode name="sistema externo"/>
<pnode name="transferência de dados"/>
</cnode>
<cnode name="legais" desc="leis específicas incidindo sobre
propriedades do sistema">
<pnode name="internacionalização"/>
</cnode>
</cnode>
<cnode name="processo" desc="características e aspectos que
restringem ou guiam o processo de desenvolvimento">
<cnode name="entrega" desc="prazos e outras características ou
propriedades necessários para a liberação do sistema">
</cnode>
<cnode name="implementação" desc="restrições ou padrões de
ambiente e tecnologia de desenvolvimento">
<cnode name="linguagens" desc="linguagens a utilizar no
desenvolvimento do projeto">
<pnode name="java"/>
<pnode name="xml"/>
</cnode>
```

```
</cnode>
<cnode name="padrões" desc="padrões de qualidade em uso na
organização">
</cnode>
</cnode>
<cnode name="produto" desc="características ou propriedades que o
sistema deverá presentar ou atender">
<cnode name="ambiente operacional" desc="ambiente operacional no
qual o software deverá executar">
<pnode name="internet"/>
<pnode name="rede local"/>
<pnode name="sistema operacional"/>
</cnode>
<cnode name="confiabilidade" desc="características relacionadas a
propriedades de execução mesmo em situações de falha">
<cnode name="integridade dos dados" desc="precisão e veracidade
das informações armazenadas">
<pnode name="transações atômicas"/>
</cnode>
<cnode name="segurança" desc="aspectos relacionados à segurança de
informações">
<pnode name="gerar log"/>
<pnode name="gerar um log"/>
<pnode name="log"/>
</cnode>
<cnode name="tolerância a falhas" desc="habilidade em manter a
operação em caso de falhas">
</cnode>
</cnode>
<cnode name="confidencialidade" desc="proteção no acesso ao
sistema e às informações manipuladas">
<cnode name="cadastrar usuários" desc="registro e manutenção de
usuários válidos">
<pnode name="activar utilizador"/>
<pnode name="ativar usuário"/>
<pnode name="controle de acesso"/>
<pnode name="manter usuário"/>
<pnode name="perfil de usuário"/>
<pnode name="perfis de usuário"/>
</cnode>
<cnode name="conectar e autenticar usuários" desc="validação de
usuários na conexão">
<pnode name="autenticação"/>
<pnode name="login"/>
<pnode name="logon"/>
<pnode name="password"/>
<pnode name="password * acesso"/>
<pnode name="senha"/>
<pnode name="senha de acesso"/>
<pnode name="validação de usuário"/>
</cnode>
<cnode name="desconectar" desc="encerramento de conexão">
<pnode name="desconectar"/>
<pnode name="encerrar sessão"/>
<pnode name="logoff"/>
<pnode name="logout"/>
</cnode>
</cnode>
<cnode name="eficiência" desc="aspectos relacionados ao grau de
recursos utilizados e tempo de processamento">
```

```
<cnode name="desempenho" desc="tempo de resposta e de
processamento">
<pnode name="tempo de execução"/>
<pnode name="tempo de resposta"/>
</cnode>
<cnode name="recursos" desc="recursos necessários para uso e
armazenamento de informações">
<pnode name="espaço em disco"/>
<pnode name="memória principal"/>
</cnode>
</cnode>
<cnode name="manutenibilidade" desc="características relacionadas
ao processo de manutenção">
<cnode name="modifiabilidade" desc="características relacionadas a
facilidade e ao  esforço necessários para modificações no
software">
</cnode>
<cnode name="testabilidade" desc="esforço necessário para a
verificação da correção">
</cnode>
</cnode>
<cnode name="portabilidade" desc="características que indicam o
grau de adaptação a diferentes plataformas">
<cnode name="adaptabilidade" desc="características relacionadas a
aspectos de utilização do software em diferentes ambientes
operacionais">
</cnode>
</cnode>
<cnode name="usabilidade" desc="características relacionadas à
aspectos de utilização do software">
<cnode name="flexibilidade" desc="características relacionadas à
adaptação da interface com usuário ">
<pnode name="adaptação ao usuário"/>
<pnode name="customização"/>
</cnode>
<cnode name="operabilidade" desc="esforço para operar e controlar
as operações">
<pnode name="facilidade de uso"/>
</cnode>
<cnode name="web" desc="características de usabilidade próprias
para sistemas web">
<pnode name="acesso via web"/>
<pnode name="navegador"/>
</cnode>
</cnode>
</cnode>
<cnode name="suporte" desc="relacionados a características de
apoio ao sistema">
<cnode name="documentação" desc="documentação necessária para a
instalação e utilização do software">
<pnode name="manual de suporte"/>
<pnode name="manual de usuário"/>
<pnode name="manual do usuário"/>
<pnode name="manual do utilizador"/>
</cnode>
<cnode name="treinamento" desc="capacitação dos usuários para a
correta operação do sistema">
</cnode>
</cnode>
</cnode>
</dictionary>
```

Apêndice D
Código dos agentes da aplicação

Agente Comunicador (communicator.java)

```
package requisitos.agents;

import java.text.SimpleDateFormat;
import java.util.List;
import java.util.Map;
import java.util.Set;

import javax.mail.MessagingException;
import javax.mail.internet.AddressException;

import org.midas.as.AgentServer;
import org.midas.as.agent.board.Board;
import org.midas.as.agent.board.Message;
import org.midas.as.agent.board.MessageListener;
import org.midas.as.agent.templates.Agent;
import org.midas.as.agent.templates.LifeCycleException;
import org.midas.as.agent.templates.ServiceException;
import org.midas.as.manager.execution.Logger;

import requisitos.business.entities.User;
import requisitos.business.entities.VerifyValidateProccess;
import requisitos.dao.RequirementDocumentDao;
import requisitos.web.email.Email;

public class Communicator extends Agent implements MessageListener
{
	private static String className = "[COMMUNICATOR AGENT]  ";
	private static SimpleDateFormat formatter = new SimpleDateFormat("dd/MM/yyyy
HH:mm");

	@Override
	protected void lifeCycle() throws LifeCycleException, InterruptedException
	{
		// Se registra no BlackBoard
		Board.addMessageListener("Communicator",this);
	}

	@Override
	public void provide(String service, Map<String, Object> in, List<Object> out) throws
ServiceException
	{

	}

	public void boardChanged(Message msg)
	{
		String title = msg.getTitle();
```

```
                if ( title.equals("PreVerificationStage") )
                {
                        String documentName = msg.getContent();

                        // Notifica os outros agentes
                        Logger.addEntry(className+"Notifying Users of PreVerificationStage
in "+documentName,true);
                        sendPreVerification(documentName);
                }
                if ( title.equals("VerificationStage") )
                {
                        String documentName = msg.getContent();

                        // Notifica os outros agentes
                        Logger.addEntry(className+"Notifying Users of VerificationStage in
"+documentName,true);
                        sendVerification(documentName);
                }
                if ( title.equals("PreValidationStage") )
                {
                        String documentName = msg.getContent();

                        // Notifica os outros agentes
                        Logger.addEntry(className+"Notifying Users of PreValidationStage
in "+documentName,true);
                        sendPreValidation(documentName);
                }
                if ( title.equals("ValidationStage") )
                {
                        String documentName = msg.getContent();

                        // Notifica os outros agentes
                        Logger.addEntry(className+"Notifying Users of ValidationStage in
"+documentName,true);
                        sendValidation(documentName);
                }
                if ( title.equals("Finalization") )
                {
                        String documentName = msg.getContent();

                        // Notifica os outros agentes
                        Logger.addEntry(className+"Notifying Users of Finalization in
"+documentName,true);
                        finalization(documentName);
                }
                if ( title.equals("SimilarityTest") )
                {
                        String documentName = msg.getContent();

                        // Notifica os outros agentes
                        Logger.addEntry(className+"Notifying Users of SimilarityTests on
"+documentName,true);
                }
        }

        private void sendPreVerification(String documentName)
        {
                RequirementDocumentDao rdao =
(RequirementDocumentDao)AgentServer.getApplicationContext().getBean("requirementDocume
```

```
ntDao");
                List<VerifyValidateProccess> vvproccessList =
rdao.getVerifyValidateProccessByDocumentName(documentName);

                VerifyValidateProccess vvproccess = vvproccessList.get(vvproccessList.size()-
1);
                Set<User> users = vvproccess.getUsers();

                for (User user : users)
                {
                        try
                        {
                                Email.EnviaEmail(user.getEmail(), "Rq.NET - Notificação
sobre o Documento "+vvproccess.getRequirementDocument().getName()+" - Pré-Verificação",
                                                "Caro
"+user.getUsername()+",\n\n" +
                                                "O documento
"+vvproccess.getRequirementDocument().getName()+" foi agendado para um processo de
verificação.\n\n"+
                                                "O dia marcado para o
início do processo é "+formatter.format(vvproccess.getPreVerification().getDeadline())+", até " +
                                                "lá o documento se
encontra no estágio de Pré-Verificação. Testes de similaridade serão rodados e enviados por "+
                                                "email assim que
concluídos.\n\n");
                        }
                        catch (AddressException e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                        } catch (MessagingException e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                        } catch (Exception e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                        }
                }
        }

        private void sendVerification(String documentName)
        {
                RequirementDocumentDao rdao =
(RequirementDocumentDao)AgentServer.getApplicationContext().getBean("requirementDocume
ntDao");
                List<VerifyValidateProccess> vvproccessList =
rdao.getVerifyValidateProccessByDocumentName(documentName);

                VerifyValidateProccess vvproccess = vvproccessList.get(vvproccessList.size()-
1);
                Set<User> users = vvproccess.getUsers();

                for (User user : users)
                {
                        try
                        {
                                Email.EnviaEmail(user.getEmail(), "Rq.NET - Notificação
sobre o Documento "+vvproccess.getRequirementDocument().getName()+" - Verificação",
                                                "Caro
"+user.getUsername()+",\n\n" +
                                                "O documento
```

```
"+vvproccess.getRequirementDocument().getName()+" se encontra a partir de agora em processo de verificação.\n\n");
                        }
                        catch (AddressException e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                        } catch (MessagingException e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                        } catch (Exception e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                        }
                }
        }

        private void sendPreValidation(String documentName)
        {
                RequirementDocumentDao rdao = (RequirementDocumentDao)AgentServer.getApplicationContext().getBean("requirementDocumentDao");
                List<VerifyValidateProccess> vvproccessList = rdao.getVerifyValidateProccessByDocumentName(documentName);

                VerifyValidateProccess vvproccess = vvproccessList.get(vvproccessList.size()-1);
                Set<User> users = vvproccess.getUsers();

                for (User user : users)
                {
                        try
                        {
                                Email.EnviaEmail(user.getEmail(), "Rq.NET - Notificação sobre o Documento "+vvproccess.getRequirementDocument().getName()+" - Pré-Validação",
                                                "Caro "+user.getUsername()+",\n\n" +
                                                "O documento "+vvproccess.getRequirementDocument().getName()+" teve seu processo de verificação concluído, e se "+
                                                "encontra no estágio de Pré-Validação. A etapa de validação deverá começar a partir do dia "+formatter.format(vvproccess.getPreValidation().getDeadline())+
                                                ".");

                        }
                        catch (AddressException e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                        } catch (MessagingException e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                        } catch (Exception e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                        }
                }
        }

        private void sendValidation(String documentName)
        {
```

```
		RequirementDocumentDao rdao = (RequirementDocumentDao)AgentServer.getApplicationContext().getBean("requirementDocumentDao");
		List<VerifyValidateProccess> vvproccessList = rdao.getVerifyValidateProccessByDocumentName(documentName);

		VerifyValidateProccess vvproccess = vvproccessList.get(vvproccessList.size()-1);
		Set<User> users = vvproccess.getUsers();

		for (User user : users)
		{
			try
			{
				Email.EnviaEmail(user.getEmail(), "Rq.NET - Notificação sobre o Documento "+vvproccess.getRequirementDocument().getName()+" - Validação",
								"Caro "+user.getUsername()+",\n\n" +
								"O documento "+vvproccess.getRequirementDocument().getName()+" se encontra a partir de agora em processo de validação.\n\n");
			}
			catch (AddressException e) {
				// TODO Auto-generated catch block
				e.printStackTrace();
			} catch (MessagingException e) {
				// TODO Auto-generated catch block
				e.printStackTrace();
			} catch (Exception e) {
				// TODO Auto-generated catch block
				e.printStackTrace();
			}
		}
	}

	private void finalization(String documentName)
	{
		RequirementDocumentDao rdao = (RequirementDocumentDao)AgentServer.getApplicationContext().getBean("requirementDocumentDao");
		List<VerifyValidateProccess> vvproccessList = rdao.getVerifyValidateProccessByDocumentName(documentName);

		VerifyValidateProccess vvproccess = vvproccessList.get(vvproccessList.size()-1);
		Set<User> users = vvproccess.getUsers();

		for (User user : users)
		{
			try
			{
				Email.EnviaEmail(user.getEmail(), "Rq.NET - Notificação sobre o Documento "+vvproccess.getRequirementDocument().getName()+" - Finalização",
								"Caro "+user.getUsername()+",\n\n" +
								"O documento "+vvproccess.getRequirementDocument().getName()+" teve seu processo de verificação ou validação finalizado. " +
								"Ele se encontra agora em aberto para modifiações e inserções.");
```

```
                        }
                        catch (AddressException e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                        } catch (MessagingException e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                        } catch (Exception e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                        }
                }
        }
}
```

Agente Analisador Léxico (lexical.java)

```
package requisitos.agents;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Set;
import java.util.TreeSet;

import org.apache.commons.lang.StringUtils;
import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;
import org.apache.regexp.RE;
import org.midas.as.agent.board.Board;
import org.midas.as.agent.board.Message;
import org.midas.as.agent.board.MessageListener;
import org.midas.as.agent.templates.Agent;
import org.midas.as.agent.templates.LifeCycleException;
import org.midas.as.agent.templates.ServiceException;
import org.xml.sax.SAXException;

import requisitos.business.entities.Requirement;
import requisitos.business.entities.Theme;
import requisitos.business.entities.ThemeExcerpts;
import requisitos.stemmer.PortugueseStemmer;
import requisitos.stemmer.Stemmer;
import edu.harvard.wcfia.yoshikonverter.Converter;

@SuppressWarnings("unchecked")
public class Lexical extends Agent implements MessageListener
{
        //
        // Log
        //

        private static final Log LOGGER = LogFactory.getLog(Lexical.class);
```

```
	//
	// Constants
	//

	private static final String STOPWORD = "StOp_WoRd";

	//
	// Agent Interface
	//

	@Override
	protected void lifeCycle() throws LifeCycleException, InterruptedException
	{
		// Se registra no BlackBoard
		Board.addMessageListener("Lexical",this);
	}

	public void boardChanged(Message msg)
	{

	}

	@Override
	public void provide(String service, Map<String, Object> in, List<Object> out) throws
ServiceException
	{
		/*
		 * ============== Serviço - extractRequirements ===================
		 *
		 * Extrai os requisitos de um documento texto, para cada requisito
		 * cria um objeto do tipo Requirement, e retorna todos em uma lista
		 *
		 *
=====================================================================
		 */
		if (service.equals("extractRequirements"))
		{
			// Entrada
			File file = (File)in.get("file");

			// Execução
			List<Requirement> requirementList = extractRequirements(file);

			// Saída
			out.add(requirementList);
		}

		/*
		 *	 ==================	Serviço	-	concordance
=======================
		 *
		 * Dado um tema e um conjunto de requisitos, realiza a concordância
		 * selecionando os trechos em que o tema aparece, junto com as cinco
		 * palavras anteriores e posteriores, utiliza a stopList como filtro
		 * de identificação para palavras irrelevantes.
		 *
		 *
=====================================================================
		 */
		else if (service.equals("concordance"))
```

```
		{
			// Entrada
			Theme theme = (Theme) in.get("theme");
			List<String> stopList = (List<String>) in.get("stopList");
			List<Requirement> requirements = (List<Requirement>)
in.get("requirements");

			// Execução
			ThemeExcerpts excerpts = concordance(theme, stopList, requirements);

			// Retorno
			out.add(excerpts);
		}

		/*
		 *	================ Serviço - requirementsRegexMatcher
==================
		 *
		 * Dado um tema e uma lista de padrões, procura no texto dos requisitos
		 * quando o tema ocorre junto a cada um dos padrões, com uma distância
		 * de 4 ou menos palavras.
		 *
		 * Retorna um mapa em que as chaves são os padrões, e a entrada para
		 * cada padrão é uma lista com os códigos dos requisitos em que o
		 * padrão ocorre junto ao tema.
		 *
		 *
=====================================================================
		 */
		else if (service.equals("requirementsRegexMatcher"))
		{
			// Entrada
			String theme = (String)in.get("theme");
			List<String> patterns = (List<String>) in.get("patterns");
			List<Requirement> requirements = (List<Requirement>)
in.get("requirements");

			// Execução
			Map<String,Set<String>> matchs = requirementsRegexMatcher(theme,
patterns, requirements);

			// Retorno
			out.add(matchs);
		}

		/*
		 *	========================= Serviço - stemmer
==========================
		 *
		 * Dado uma palavra, ou um conjunto de palavras, este serviço retorna
		 * o(s) radicai(s) correspondentes.
		 *
		 *
=====================================================================
		 */
		else if (service.equals("stemmer"))
		{
			// Entrada
			String word = (String)in.get("word");
```

```
                    // Execução
                    Stemmer stem                = new PortugueseStemmer();

                    String  steemedWord = stem.stem(word);

                    // Retorno
                    out.add(steemedWord);
                }

                /*
                 *      ======================      Serviço     -     convertPdf
========================
                 *
                 * Lê o conteúdo de um arquivo pdf, e salva em um novo arquivo texto.
                 * Utiliza a biblioteca YoshikoderConverter
                 *
                 *
=============================================================================
                 */
                else if (service.equals("convertPdf"))
                {
                    // Entrada
                    File pdfFile = (File)in.get("pdfFile");
                    File txtFile = (File)in.get("txtFile");

                    try
                    {
                        // Execução
                        String text = Converter.inhalePdf(pdfFile);
                  dumpText(txtFile, text);
                    }
                    catch (IOException e)
                    {
                        throw new ServiceException("Unable to write text file -
"+txtFile.getAbsolutePath(),e);
                    }
                }

                /*
                 *      ======================      Serviço     -     convertDoc
========================
                 *
                 * Lê o conteúdo de um arquivo doc, e salva em um novo arquivo texto.
                 * Utiliza a biblioteca YoshikoderConverter
                 *
                 *
=============================================================================
                 */
                else if (service.equals("convertDoc"))
                {
                    // Entrada
                    File docFile = (File)in.get("docFile");
                    File txtFile = (File)in.get("txtFile");

                    try
                    {
                        // Execução
                        String text = Converter.inhaleDoc(docFile);
                  dumpText(txtFile, text);
                    }
```

```
                        catch (IOException e)
                        {
                                throw new ServiceException("Unable to write text file -
"+txtFile.getAbsolutePath(),e);
                        }
                }

                /*
                 *      ======================      Serviço     -     convertHtml
=========================
                 *
                 * Lê o conteúdo de um arquivo html, e salva em um novo arquivo texto.
                 * Utiliza a biblioteca YoshikoderConverter
                 *
                 *
=========================================================================
                 */
                else if (service.equals("convertHtml"))
                {
                        // Entrada
                        File htmlFile = (File)in.get("htmlFile");
                        File txtFile = (File)in.get("txtFile");

                        try
                        {
                                // Execução
                                String                  text                    =
Converter.inhaleHtml("file:\\"+htmlFile.getAbsolutePath());
                            dumpText(txtFile, text);
                        }
                        catch (SAXException e)
                        {
                                throw new ServiceException("Unable to read html file -
"+htmlFile.getAbsolutePath(),e);
                        }
                        catch (IOException e)
                        {
                                throw new ServiceException("Unable to write text file -
"+txtFile.getAbsolutePath(),e);
                        }
                }
        }

        //
        //  Methods
        //

        public static List<Requirement> extractRequirements(File fullFile)
        {
                LOGGER.info("Extracting Requirements");

                String line;

                try
                {
                        LOGGER.debug("Opening read and write buffers");

                        String          fullFilePath                            =
fullFile.getAbsolutePath().replaceAll(fullFile.getName(),"")+"//files//";
                        File   fullFilePathFile = new File(fullFilePath);
```

```
fullFilePathFile.mkdir();

BufferedReader br = new BufferedReader(new FileReader(fullFile));

Requirement requirement = null;
List<Requirement> requirementList = new ArrayList<Requirement>();

boolean firstLine = true;
boolean writing = false;

// Lê uma Linha
while ( (line = br.readLine()) != null)
{
	line = line.trim();

	// SE é vazia
	if (line.equals("") || firstLine)
	{
		// Pega o provável título do próximo arquivo
		String title;

		if (firstLine)
		{
			title = line;
			firstLine = false;
		}
		else
		{
			title = br.readLine();
		}


		// SE é um título
		if (!title.equals("") && (title.contains("RF") ||
title.contains("RNF")) )
		{
			// Remove qualquer lixo antes do título
			if (title.contains("RF"))
			{

				while (!title.startsWith("RF"))
					title = title.substring(1);

			}
			else
			{
				while (!title.startsWith("RNF"))
					title = title.substring(1);
			}


			// Troca barras por traços
			title.replaceAll("/","-");

			// Separa código do título
			String[] titleSplit = title.split(" ",2);

			// Cria atributos do requisito
			String reqCode = titleSplit[0];
```

```
                                        String reqTitle = titleSplit[1];

                                        writing = true;

                                        // Adiciona o requisito na lista de saída
                                        requirement = new
Requirement(reqCode,reqTitle);
                                        requirementList.add(requirement);

                                        LOGGER.debug("Requirement Extracted:
"+title);
                                }
                                // SENÃO
                                else
                                {
                                        // Não cria o arquivo
                                        writing = false;
                                }
                        }
                        // SENÃO
                        else
                        {
                                // Caso seja possível adiciona o texto ao arquivo
criado
                                if (writing)
                                {
                                        requirement.addText(line+"\n");
                                }
                        }
                }

                return requirementList;
        }
        catch(IOException e)
        {
                e.printStackTrace();
                return null;
        }
    }

    public static ThemeExcerpts concordance(Theme theme,List<String>
stopList,List<Requirement> requirements)
    {
        String themeName = theme.getName();
        ThemeExcerpts excerpts = new ThemeExcerpts(theme.getName());

        LOGGER.debug("Processando concordancia nos requisitos para o tema:
"+themeName);

        for (Requirement requirement : requirements)
        {
                LOGGER.debug("Requisito: "+requirement.getCode());

                String text = requirement.getText();
                String[] textParagraphs = StringUtils.split(text,"\n");

                for (String paragraph : textParagraphs)
                {
                        String[] textPhrases =
StringUtils.splitByWholeSeparator(paragraph,". ");
```

```
                    for (String phrase : textPhrases)
                    {
                        // Limpa a frase
                        String cleanText = cleanToken(phrase,stopList);

                        // Stemmiza a frase
                        Stemmer stemmer = new PortugueseStemmer();
                        String stemmizedText =
stemmer.stemString(cleanText);

                        // SE a linha contém a expressão OU o pós-buffer não
está vazio
                        int expressionCount =
StringUtils.countMatches(stemmizedText,themeName);

                        for (int k = 0; k < expressionCount; k++)

                        {
                            // Quebra o parágrafo pelo token
                            String[] paragraphPieceArray =
StringUtils.splitByWholeSeparator(stemmizedText,themeName);

                            StringBuilder prePiece = new
StringBuilder();
                            StringBuilder posPiece = new
StringBuilder();

                            for (int i = 0 ; i <= k; i++)
                            {
                                prePiece.append("
"+paragraphPieceArray[i].trim());

                                if (i < k)
                                    prePiece.append("
"+themeName);
                            }

                            for (int i = k+1 ; i <
paragraphPieceArray.length ; i++)
                            {
                                posPiece.append("
"+paragraphPieceArray[i].trim());

                                if (i < paragraphPieceArray.length-
1)
                                    posPiece.append("
"+themeName);
                            }

                            // Recupera o pedaço atual
                            String excerptPrePiece =
prePiece.toString().trim();
                            String[] excerptPrePieceSplit =
excerptPrePiece.split(" ");

                            // Recupera o próximo pedaço
                            String excerptPosPiece =
posPiece.toString().trim();
                            String[] excerptPosPieceSplit =
```

```
excerptPosPiece.split(" ");

                                                // Monta a extração
                                                StringBuilder excerptBuilder = new
StringBuilder();

                                                if (excerptPrePieceSplit.length>=5)
excerptBuilder.append(excerptPrePieceSplit[excerptPrePieceSplit.length-5]+" ");
                                                if (excerptPrePieceSplit.length>=4)
excerptBuilder.append(excerptPrePieceSplit[excerptPrePieceSplit.length-4]+" ");
                                                if (excerptPrePieceSplit.length>=3)
excerptBuilder.append(excerptPrePieceSplit[excerptPrePieceSplit.length-3]+" ");
                                                if (excerptPrePieceSplit.length>=2)
excerptBuilder.append(excerptPrePieceSplit[excerptPrePieceSplit.length-2]+" ");
                                                if (excerptPrePieceSplit.length>=1)
excerptBuilder.append(excerptPrePieceSplit[excerptPrePieceSplit.length-1]+" ");

                                                excerptBuilder.append(themeName+" ");

                                                if (excerptPosPieceSplit.length>=1)
excerptBuilder.append(excerptPosPieceSplit[0]+" ");
                                                if (excerptPosPieceSplit.length>=2)
excerptBuilder.append(excerptPosPieceSplit[1]+" ");
                                                if (excerptPosPieceSplit.length>=3)
excerptBuilder.append(excerptPosPieceSplit[2]+" ");
                                                if (excerptPosPieceSplit.length>=4)
excerptBuilder.append(excerptPosPieceSplit[3]+" ");
                                                if (excerptPosPieceSplit.length>=5)
excerptBuilder.append(excerptPosPieceSplit[4]+" ");

                                                String excerpt = excerptBuilder.toString();

                                                excerpts.addExcerpt(requirement,excerpt);


                                                LOGGER.debug("Extração: "+excerpt);
                                        }
                                }
                        }
                }

                return excerpts;
        }

        public static Map<String,Set<String>> requirementsRegexMatcher(String theme,
List<String> patterns, List<Requirement> requirements)
        {
                Map<String,String>          requirementTexts          =          new
HashMap<String,String>();
                Map<String,Set<String>> requirementAssociations = new HashMap<String,
Set<String>>();

                theme = new PortugueseStemmer().stem(theme);

                try
                {
                        LOGGER.info("Procurando associações entre os padrões e os
requisitos:");

                        // Recupera o texto de cada requisito
```

```
                        for (Requirement requirement : requirements)
                        {

        requirementTexts.put(requirement.getCode(),requirement.getText());
                        }

                        // Cruza cada padrão contra os textos recuperados
                        for (String pattern : patterns)
                        {
                                LOGGER.debug("Padrão: "+pattern);

                                RE regex1 = new
RE("("+theme+"[:alnum:]*){1}(([:blank:]+[:alnum:]+){1,4})?([:blank:]+)("+pattern+"[:alnum:]*)
{1}");
                                RE regex2 = new
RE("("+pattern+"[:alnum:]*){1}(([:blank:]+[:alnum:]+){1,4})?([:blank:]+)("+theme+"[:alnum:]*)
{1}");

                                Set<String> reList = new TreeSet<String>();

                                requirementAssociations.put(pattern,reList);

                                for (Requirement requirement : requirements)
                                {
                                        // Recupera o texto, joga todas as letras para
                                        // minúsculas e limpa acentos
                                        String text =
requirementTexts.get(requirement.getCode());
                                        text = uniformizeText(text);

                                        // Find all the matches.
                                        if(regex1.match(text))
                                        {
                                                int count = regex1.split(text).length-1;

                                                LOGGER.debug("Encontrado "+count+"
vezes em: "+requirement.getCode());
                                                reList.add(requirement.getCode());
                                        }
                                        else if(regex2.match(text))
                                        {
                                                int count = regex2.split(text).length-1;

                                                LOGGER.debug("Encontrado "+count+"
vezes em: "+requirement.getCode());
                                                reList.add(requirement.getCode());
                                        }

                                }
                        }

                        return requirementAssociations;
                }
                catch(Exception e)
                {
                        e.printStackTrace();
                        return null;
                }
        }
```

```
		private static String uniformizeText(String text)
		{
			String fragment = StringUtils.lowerCase(text);

			fragment = fragment.replace('à','a');
			fragment = fragment.replace('á','a');
			fragment = fragment.replace('é','e');
			fragment = fragment.replace('í','i');
			fragment = fragment.replace('ó','o');
			fragment = fragment.replace('ú','u');
			fragment = fragment.replace('ã','a');
			fragment = fragment.replace('õ','o');
			fragment = fragment.replace('â','a');
			fragment = fragment.replace('ê','e');
			fragment = fragment.replace('ô','o');
			fragment = fragment.replace('ü','u');
			fragment = fragment.replace(',',' ');
			fragment = fragment.replace('.',' ');
			fragment = fragment.replace(';',' ');

			return fragment;
		}

		private static String cleanToken(String token, List<String> stopList)
		{
			String temp = token;

			temp = StringUtils.stripStart(temp," ");
			temp = StringUtils.stripEnd(temp," ");
			temp = StringUtils.remove(temp," ┌ ");
			temp = StringUtils.remove(temp,",");
			temp = StringUtils.remove(temp,".");
			temp = StringUtils.remove(temp,":");
			temp = StringUtils.remove(temp,";");
			temp = StringUtils.remove(temp,"[");
			temp = StringUtils.remove(temp,"]");
			temp = StringUtils.remove(temp,"{");
			temp = StringUtils.remove(temp,"}");
			temp = StringUtils.remove(temp,"(");
			temp = StringUtils.remove(temp,")");
			temp = StringUtils.remove(temp,"!");
			temp = StringUtils.remove(temp,"?");
			temp = StringUtils.remove(temp,"/");
			temp = StringUtils.remove(temp,"'");
			temp = StringUtils.remove(temp,"<");
			temp = StringUtils.remove(temp,">");
			temp = StringUtils.remove(temp,"*");
			temp = StringUtils.remove(temp,"\"");
			temp = StringUtils.remove(temp,"\\");
			temp = StringUtils.lowerCase(temp);

			for (String stopWord : stopList)
			{
				temp = StringUtils.replace(temp," "+stopWord+" "," "+STOPWORD+"
");
			}

			return temp;
		}
```

```
	private void dumpText(File txtFile, String text) throws IOException
	{
		BufferedWriter writer = new BufferedWriter(new FileWriter(txtFile));
		writer.write(text);
		writer.close();
	}
}
```

Agente Construtor do Léxico (lexicalConstructor.java)

```
package requisitos.agents;

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.util.List;
import java.util.Map;

import org.midas.as.agent.board.Board;
import org.midas.as.agent.board.Message;
import org.midas.as.agent.board.MessageListener;
import org.midas.as.agent.templates.Agent;
import org.midas.as.agent.templates.LifeCycleException;
import org.midas.as.agent.templates.ServiceException;

import corpus.FileTokeniser;

public class LexicalConstructor extends Agent implements MessageListener
{
	@Override
	protected void lifeCycle() throws LifeCycleException, InterruptedException
	{
		// Se registra no BlackBoard
		Board.addMessageListener("LexicalConstructor",this);
	}

	@Override
	public void provide(String service, Map<String, Object> in, List<Object> out) throws
ServiceException
	{
		if (service.equals("tagger"))
		{

		}
		/*
		 *	=====================	Serviço	-	tokenizer
=========================
		 *
		 *
		 *
		 *
=========================================================================
		 */
		else if (service.equals("tokenizer"))
		{
			// Entrada
			File srcFile = (File)in.get("srcFile");
			File txtFile = (File)in.get("txtFile");
```

```
                        try
                        {
                                FileTokeniser ft = new
FileTokeniser(srcFile.getAbsolutePath());
                                BufferedWriter writer = new BufferedWriter(new
FileWriter(txtFile));

                                while (ft.hasMoreTokens())
                                {
                                        writer.write(ft.getNextToken()+"\n");
                                }

                                writer.close();
                        }
                        catch (IOException e)
                        {
                                throw new ServiceException("Unable to write text file -
"+txtFile.getAbsolutePath(),e);
                        }
                }
        }

        public void boardChanged(Message msg)
        {

        }
}
```

Agente Manager (manager.java)

```
package requisitos.agents;

import java.util.List;
import java.util.Map;

import org.midas.as.agent.board.Board;
import org.midas.as.agent.board.BoardException;
import org.midas.as.agent.board.Message;
import org.midas.as.agent.board.MessageListener;
import org.midas.as.agent.templates.Agent;
import org.midas.as.agent.templates.LifeCycleException;
import org.midas.as.agent.templates.ServiceException;
import org.midas.as.manager.execution.Logger;

public class Manager extends Agent implements MessageListener
{
        private static String className = "[MANAGER AGENT]  ";

        @Override
        protected void lifeCycle() throws LifeCycleException, InterruptedException
        {
                // Se registra no BlackBoard
                Board.addMessageListener("Manager",this);
        }

        @Override
        public void provide(String service, Map<String, Object> in, List<Object> out) throws
ServiceException
        {
```

```
	}

	public void boardChanged(Message msg)
	{
		String title = msg.getTitle();

		if ( title.equals("PreVerificationStage") )
		{
			String documentName = msg.getContent();

			// Notifica os outros agentes
			Logger.addEntry(className+"Document "+msg.getContent()+" is
under PreVerificationStage, notifiyng other Agents...",true);

			try
			{

	Board.writeOnBoard(1,"Communicator","PreVerificationStage",documentName,this);

	Board.writeOnBoard(1,"Verifier","PreVerificationStage",documentName,this);
			}
			catch (BoardException e)
			{
				e.printStackTrace();
			}
		}
		if ( title.equals("VerificationStage") )
		{
			String documentName = msg.getContent();

			// Notifica os outros agentes
			Logger.addEntry(className+"Document "+msg.getContent()+" is
under VerificationStage, notifiyng other Agents...",true);

			try
			{

	Board.writeOnBoard(1,"Communicator","VerificationStage",documentName,this);

			}
			catch (BoardException e)
			{
				e.printStackTrace();
			}
		}
		if ( title.equals("PreValidationStage") )
		{
			String documentName = msg.getContent();

			// Notifica os outros agentes
			Logger.addEntry(className+"Document "+msg.getContent()+" is
under PreValidationStage, notifiyng other Agents...",true);

			try
			{

	Board.writeOnBoard(1,"Communicator","PreValidationStage",documentName,this);

			}
			catch (BoardException e)
```

```
                        {
                                e.printStackTrace();
                        }
                }
                if ( title.equals("ValidationStage") )
                {
                        String documentName = msg.getContent();

                        // Notifica os outros agentes
                        Logger.addEntry(className+"Document     "+msg.getContent()+"     is
under ValidationStage, notifiyng other Agents...",true);

                        try
                        {

        Board.writeOnBoard(1,"Communicator","ValidationStage",documentName,this);

                        }
                        catch (BoardException e)
                        {
                                e.printStackTrace();
                        }
                }
                if ( title.equals("Finalization") )
                {
                        String documentName = msg.getContent();

                        // Notifica os outros agentes
                        Logger.addEntry(className+"Document    "+msg.getContent()+"    has
been released, notifiyng other Agents...",true);

                        try
                        {

        Board.writeOnBoard(1,"Communicator","Finalization",documentName,this);

                        }
                        catch (BoardException e)
                        {
                                e.printStackTrace();
                        }
                }
        }
}
```

Agente Observador (observer.java)

```
package requisitos.agents;

import java.util.List;
import java.util.Map;

import org.midas.as.agent.board.Board;
import org.midas.as.agent.board.Message;
import org.midas.as.agent.board.MessageListener;
import org.midas.as.agent.templates.Agent;
import org.midas.as.agent.templates.LifeCycleException;
import org.midas.as.agent.templates.ServiceException;
import org.midas.as.manager.execution.Logger;

public class Observer extends Agent implements MessageListener
```

```
{
	private static String className = "[OBSERVER AGENT]  ";

	@Override
	protected void lifeCycle() throws LifeCycleException, InterruptedException
	{
		// Se registra no BlackBoard
		Board.addMessageListener("Observer",this);
	}

	@Override
	public void provide(String service, Map<String, Object> in, List<Object> out) throws
ServiceException
	{

	}

	public void boardChanged(Message msg)
	{
		String title = msg.getTitle();

		if ( title.equals("Document Created") ||
			 title.equals("Document Removed") ||
			 title.equals("Document Altered")  )
		{
			// Notifica os outros agentes
			Logger.addEntry(className+msg.getContent(),true);
		}

//		// Se o documento de requisitos foi alterado
//		if (message.equals("Document Altered"))
//		{
//			// Notifica os outros agentes
//			Board.writeForAll("Document Altered",this);
//		}
//
//		// Se um documento de requisitos foi criado
//		else if (message.equals("Document Created"))
//		{
//			// Notifica os outros agentes
//			Board.writeForAll("Document Created",this);
//		}
//
//		// Se o documento de requisitos foi removido
//		else if (message.equals("Document Removed"))
//		{
//			// Notifica os outros agentes
//			Board.writeForAll("Document Removed",this);
//		}
	}
}
```

Agente Estatístico (statistical.java)

```
package requisitos.agents;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Set;
```

```
import java.util.Map.Entry;

import org.apache.commons.lang.StringUtils;
import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;
import org.midas.as.agent.board.Board;
import org.midas.as.agent.board.Message;
import org.midas.as.agent.board.MessageListener;
import org.midas.as.agent.templates.Agent;
import org.midas.as.agent.templates.LifeCycleException;
import org.midas.as.agent.templates.ServiceException;

import requisitos.business.entities.FrequencyMap;
import requisitos.business.entities.Requirement;
import requisitos.business.entities.ThemeExcerpts;
import requisitos.business.entities.ThemeMetrics;

@SuppressWarnings("unchecked")
public class Statistics extends Agent implements MessageListener
{
	//
	// Log
	//

	private static final Log LOGGER = LogFactory.getLog(Statistics.class);

	//
	// Constants
	//

	private static final String STOPWORD = "stop_word";

	//
	// Agent Interface
	//

	@Override
	protected void lifeCycle() throws LifeCycleException, InterruptedException
	{
		// Se registra no BlackBoard
		Board.addMessageListener("Statistics",this);
	}

	public void boardChanged(Message msg)
	{

	}

	@Override
	public void provide(String service, Map<String, Object> in, List<Object> out) throws
ServiceException
	{
		/*
		 *	==================	Serviço	-	frequencyMap
========================
		 *
		 * Dadas as extrações do Concordanciador e o texto completo com os
		 * requisitos, o serviço de frequencyMap calcula o Mapa de Frequência
		 * completo para o tema.
		 *
```

```
                *
================================================================
                */
                if (service.equals("frequencyMap"))
                {
                        // Entrada
                        ThemeExcerpts themeExcerpts = (ThemeExcerpts)
in.get("themeExcerpts");
                        List<Requirement> requirements = (List<Requirement>)
in.get("requirements");

                        // Execução
                        FrequencyMap frequencyMap = frequencyMap(themeExcerpts,
requirements);

                        // Saída
                        out.add(frequencyMap);
                }

                /*
                *       ====================== Serviço - metrics
==========================
                *
                * Dado o mapa de frequência, o serviço de metrics calcula os valores
                * de scoreT e informaçãoMútua para todos os termos que ocorreram
                * junto ao tema.
                *
                *
================================================================
                */
                else if (service.equals("metrics"))
                {
                        // Entrada
                        FrequencyMap frequencyMap = (FrequencyMap)
in.get("frequencyMap");

                        // Execução
                        ThemeMetrics themeMetrics = metrics(frequencyMap);

                        // Saída
                        out.add(themeMetrics);
                }
        }

        //
        // Methods
        //

        public static FrequencyMap frequencyMap(ThemeExcerpts
themeExcerpts,List<Requirement> requirements)
        {
                LOGGER.info("Calculando mapa de frequência");

                //
                // Saídas
                //

                int corpusCount = 0;
                                                        // N
                int tokenFrequency = 0;
```

```
                                                    // f(n)
                Map<String,Integer> nodesOcurrenceMap = new HashMap<String,Integer>();
        // f(c)[]
                Map<String,Integer>             nodesCoOcurrenceMap             =             new
HashMap<String,Integer>();          // f(n,c)[]
                Map<String,List<String>>          requirementOcurrenceMap          =          new
HashMap<String,List<String>>();

                //
                // Calcula f(n) e f(n,c)
                //

                tokenFrequency = themeExcerpts.getAllExcerpts().size();

                for (String code : themeExcerpts.getRequirementCodes())
                {
                        LOGGER.debug("Requisito: "+code);

                        // CO-OCURRENCE
                        Map<String,Integer>       reqNodesCoOcurrenceMap       =       new
HashMap<String,Integer>();          // f(n,c)[]

                        for (String excerpt : themeExcerpts.getExcerpts(code))
                        {
                                String[]                    tokens      = excerpt.split(" ");
                                Map<String,Boolean>          goneTokens          =          new
HashMap<String,Boolean>();

                                for (int i = 0; i < tokens.length; i++)
                                {
                                        String token = cleanToken(tokens[i]);

                                        if              (isTokenValid(token,true)              &&
!token.equals(themeExcerpts.getTheme()))
                                        {
                                                if
(reqNodesCoOcurrenceMap.containsKey(token))
                                                {

                                                        if
(!goneTokens.containsKey(token))
                                                        {

        reqNodesCoOcurrenceMap.put(token,reqNodesCoOcurrenceMap.get(token) + 1);

        goneTokens.put(token,true);
                                                        }
                                                }
                                                else
                                                {


        reqNodesCoOcurrenceMap.put(token,1);
                                                        goneTokens.put(token,true);
                                                }

                                                if
(requirementOcurrenceMap.containsKey(token))
                                                {
                                                        List<String>            reqs            =
```

```
requirementOcurrenceMap.get(token);
                                                        reqs.add(","+code);

                                                }

                                                else
                                                {
                                                        List<String>    reqs    =    new
ArrayList<String>();

        requirementOcurrenceMap.put(token,reqs);

                                                        reqs.add(code);
                                                }
                                        }
                                }
                        }

                        for             (Entry<String,Integer>          entry           :
reqNodesCoOcurrenceMap.entrySet())
                        {
                                LOGGER.debug(entry.getKey()+": "+entry.getValue());
                        }

                        for             (Entry<String,Integer>          entry           :
reqNodesCoOcurrenceMap.entrySet())
                        {
                                if (nodesCoOcurrenceMap.containsKey(entry.getKey()))
                                {
                                        Integer    newValue    =    entry.getValue()    +
nodesCoOcurrenceMap.get(entry.getKey());

        nodesCoOcurrenceMap.put(entry.getKey(),newValue);
                                }
                                else
                                {

        nodesCoOcurrenceMap.put(entry.getKey(),entry.getValue());

                                }
                        }
                }

                //
                // Calcula f(c) e N
                //

                LOGGER.debug("No texto todo: ");

                String text = getText(requirements);

                Set<String> nodesSet = nodesCoOcurrenceMap.keySet();
                String[] tokens = StringUtils.split(text," ");

                corpusCount = tokens.length;

                for (String node : nodesSet)
                {
                        int count = StringUtils.countMatches(text,node);
```

```
                    LOGGER.debug(node+": "+count);
                    nodesOcurrenceMap.put(node,count);

                }

                //
                // Adiciona na saida
                //

                FrequencyMap frequencyMap = new FrequencyMap();

                frequencyMap.setCorpusCount(corpusCount);
                frequencyMap.setTokenFrequency(tokenFrequency);
                frequencyMap.setNodesOcurrenceMap(nodesOcurrenceMap);
                frequencyMap.setNodesCoOcurrenceMap(nodesCoOcurrenceMap);

                return frequencyMap;
        }

        public static ThemeMetrics metrics (FrequencyMap frequencyMap)
        {
                //
                // Entradas
                //

                int corpusCount = frequencyMap.getCorpusCount();
                int tokenFrequency = frequencyMap.getTokenFrequency();
                Map<String,Integer>                nodesOcurrenceMap                =
frequencyMap.getNodesOcurrenceMap();
                Map<String,Integer>                nodesCoOcurrenceMap                =
frequencyMap.getNodesCoOcurrenceMap();

                Set<String> nodesOcurrenceSet = nodesOcurrenceMap.keySet();

                //
                // Saídas
                //

                Map<String,Double> tScoreMap = new HashMap<String,Double>();
// scoreT[]
                Map<String,Double>        mutuosInformationMap        =        new
HashMap<String,Double>(); // informacaoMutua[]

                for (String node: nodesOcurrenceSet)
                {
                        int nodeOcurrence = nodesOcurrenceMap.get(node);   // f(c)
                        int nodeCoOcurrence = nodesCoOcurrenceMap.get(node); // f(n,c)

                        //
                        // scoreT = ( f(n,c) – Y ) / X
                        //
                        // X = f(n,c)^1/2
                        // Y = ( f(n) f(c) / N )
                        //

                        // Calcula X = f(n,c)^1/2
                        double x = (double)Math.sqrt(nodeCoOcurrence);

                        // Calcula Y = ( f(n) f(c) / N )
                        double y = (double)tokenFrequency * nodeOcurrence / corpusCount;
```

```
                                        // Calcula scoreT = ( f(n,c) – Y ) / X
                                        double scoreT = (double)(nodeCoOcurrence-y)/x;

                                        //
                                        // mutuosInformation = log2 [ A / B ]
                                        //
                                        // A = (f(n,c) / N)
                                        // B = (f(n)/N) * (f(c)/ N)
                                        //

                                        // Calcula A = (f(n,c) / N)
                                        double a = (double)nodeCoOcurrence/corpusCount;

                                        // Calcula B = (f(n)/N) * (f(c)/ N)
                                        double b =
((double)tokenFrequency/corpusCount)*((double)nodeOcurrence/corpusCount);

                                        // Calcula mutuosInformation = log2 [ A / B ]
                                        double mutuosInformation = Math.log(a/b)/Math.log(2.0);

                                        //
                                        // Adiciona nos mapas
                                        //

                                        tScoreMap.put(node,scoreT);
                                        mutuosInformationMap.put(node,mutuosInformation);

                                }

                                ThemeMetrics metrics = new ThemeMetrics();

                                metrics.setTScoreMap(tScoreMap);
                                metrics.setMutuosInformationMap(mutuosInformationMap);

                                return metrics;
                        }

                        private static String getText(List<Requirement> requirements)
                        {
                                StringBuilder textBuilder = new StringBuilder();

                                for (Requirement requirement : requirements)
                                {
                                        String fragment = StringUtils.lowerCase(requirement.getText());

                                        fragment = fragment.replace('à','a');
                                        fragment = fragment.replace('á','a');
                                        fragment = fragment.replace('é','e');
                                        fragment = fragment.replace('í','i');
                                        fragment = fragment.replace('ó','o');
                                        fragment = fragment.replace('ú','u');
                                        fragment = fragment.replace('ã','a');
                                        fragment = fragment.replace('õ','o');
                                        fragment = fragment.replace('â','a');
                                        fragment = fragment.replace('ê','e');
                                        fragment = fragment.replace('ô','o');
                                        fragment = fragment.replace('ü','u');

                                        textBuilder.append(fragment+"\n");
```

```
		}

		return textBuilder.toString();
	}

	private static boolean isTokenValid(String token,boolean caseSensitive)
	{
		if (caseSensitive)
			token = StringUtils.lowerCase(token);

		if (token.equals(STOPWORD))
			return false;
		if (token.startsWith("rf"))
			return false;
		if (token.startsWith("rnf"))
			return false;
		if (token.equals("-"))
			return false;
		if (token.startsWith("&"))
			return false;
		if (token.equals(""))
			return false;
		if (token.equals(" "))
			return false;
		if (token.startsWith(" "))
			return false;
		if (token.startsWith("="))
			return false;
		if (token.endsWith(" "))
			return false;

		return true;
	}

	private static String cleanToken(String token)
	{
		String temp = token;

		temp = StringUtils.trim(temp);
		temp = StringUtils.remove(temp,",");
		temp = StringUtils.remove(temp,".");
		temp = StringUtils.remove(temp,":");
		temp = StringUtils.remove(temp,";");
		temp = StringUtils.remove(temp,"[");
		temp = StringUtils.remove(temp,"]");
		temp = StringUtils.remove(temp,"{");
		temp = StringUtils.remove(temp,"}");
		temp = StringUtils.remove(temp,"(");
		temp = StringUtils.remove(temp,")");
		temp = StringUtils.remove(temp,"!");
		temp = StringUtils.remove(temp,"?");
		temp = StringUtils.remove(temp,"/");
		temp = StringUtils.remove(temp,"'");
		temp = StringUtils.remove(temp,"<");
		temp = StringUtils.remove(temp,">");
		temp = StringUtils.remove(temp,"+");
		temp = StringUtils.remove(temp,"\"");
		temp = StringUtils.remove(temp,"\\");

		return temp;
```

```
	}
}
```

Agente Gerador de Visões (mapper.java)

```
package requisitos.agents;

import java.io.ByteArrayOutputStream;
import java.util.Collection;
import java.util.List;
import java.util.Map;
import java.util.Set;

import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;
import org.midas.as.agent.board.Board;
import org.midas.as.agent.board.Message;
import org.midas.as.agent.board.MessageListener;
import org.midas.as.agent.templates.Agent;
import org.midas.as.agent.templates.LifeCycleException;
import org.midas.as.agent.templates.ServiceException;
import org.tmapi.core.Association;
import org.tmapi.core.Locator;
import org.tmapi.core.TMAPIException;
import org.tmapi.core.Topic;
import org.tmapi.core.TopicMap;
import org.tmapi.core.TopicMapSystem;
import org.tmapi.core.TopicMapSystemFactory;
import org.tmapi.index.core.TopicsIndex;
import org.tmapiutils.impexp.xtm.XTMSerializer;

import requisitos.business.entities.Requirement;
import requisitos.stemmer.PortugueseStemmer;

@SuppressWarnings("unchecked")
public class TopicMapper extends Agent implements MessageListener
{
	//
	// Log
	//

	private static final Log LOGGER = LogFactory.getLog(TopicMapper.class);

	//
	// Agent Interface
	//

	@Override
	protected void lifeCycle() throws LifeCycleException, InterruptedException
	{
		// Se registra no BlackBoard
		Board.addMessageListener("Lexical",this);
	}

	public void boardChanged(Message msg)
	{

	}

	@Override
```

```
	public void provide(String service, Map<String, Object> in, List<Object> out) throws ServiceException
	{
		/*
		 *	======================	Serviço	-	build	============================
		 *
		 * Este serviço cria um mapa de tópicos a partir de uma lista de
		 * requisitos.
		 *
		 *
		 ============================================================================
		 */
		if (service.equals("build"))
		{
			// Entrada
			List<Requirement>	requirements	=	(List<Requirement>) in.get("requirements");

			// Execução
			TopicMap topicMap = build(requirements);

			// Saída
			out.add(topicMap);
		}

		/*
		 *	==================	Serviço	-	setAssociations	======================
		 *
		 * Este serviço cria as associações entre os pares tema/termo e os
		 * requisitos, tem como entrada o tema, o mapa de tópicos com os
		 * requisitos, e o mapa de associações que indica para cada termo que
		 * ocorre junto ao tema a lista de requisitos relacionada.
		 *
		 *
		 ============================================================================
		 */
		else if (service.equals("setAssociations"))
		{
			// Entrada
			String theme = (String) in.get("theme");
			TopicMap topicMap = (TopicMap) in.get("topicMap");
			Map<String,Set<String>>	associationsMap	=	(Map<String, Set<String>>) in.get("associationsMap");

			// Execução
			TopicMap	newTopicMap	=	setAssociations(topicMap, associationsMap, theme);

			// Saída
			out.add(newTopicMap);
		}
	}


	//
	// Methods
	//
```

```
	public static TopicMap build(List<Requirement> requirements)
	{
		LOGGER.info("Mapping Requirements...");

		try
		{
			////////////////////////////////////////////////////////
			// 1. Cria Mapa Vazio
				//
			////////////////////////////////////////////////////////

			LOGGER.debug("Make empty topic map");

			TopicMapSystemFactory tSystemFactory =
TopicMapSystemFactory.newInstance();
			TopicMapSystem tSystem =
	tSystemFactory.newTopicMapSystem();
			TopicMap topicMap
	=	tSystem.createTopicMap("http://www.tmapi.org/examples/exampletm");

			////////////////////////////////////////////////////////
			// 2. Inicializa Tópicos Básicos
		//
			////////////////////////////////////////////////////////

			LOGGER.debug("Make basic topics");

			Topic baseRequirementTopic = topicMap.createTopic();

			Topic inverseRequirementTopic = topicMap.createTopic();
			Topic functionalRequirementTopic = topicMap.createTopic();
			Topic nonFunctionalRequirementTopic = topicMap.createTopic();

			baseRequirementTopic.createTopicName("Requisito",null);

	baseRequirementTopic.addSubjectIdentifier(topicMap.createLocator("locator:/baseRequi
rement"));

			inverseRequirementTopic.addType(baseRequirementTopic);
			inverseRequirementTopic.createTopicName("Requisito Inverso",null);

	inverseRequirementTopic.addSubjectIdentifier(topicMap.createLocator("locator:/baseRe
quirement/inverseRequirement"));

			functionalRequirementTopic.addType(baseRequirementTopic);
			functionalRequirementTopic.createTopicName("Requisito
Funcional",null);

	functionalRequirementTopic.addSubjectIdentifier(topicMap.createLocator("locator:/base
Requirement/functionalRequirement"));

			nonFunctionalRequirementTopic.addType(baseRequirementTopic);
			nonFunctionalRequirementTopic.createTopicName("Requisito Não
Funcional",null);

	nonFunctionalRequirementTopic.addSubjectIdentifier(topicMap.createLocator("locator:/
baseRequirement/nonfunctionalRequirement"));

			Topic textOccurrenceTopic = topicMap.createTopic();
			textOccurrenceTopic.createTopicName("Texto",null);
```

```
		textOccurrenceTopic.addSubjectIdentifier(topicMap.createLocator("locator:/baseRequire
ment/textOcurrence"));

				//////////////////////////////////////////////////////////
				// 3. Cria tópicos dos requisitos
			//
				//////////////////////////////////////////////////////////

				LOGGER.debug("Make extracted requirements topics");

				// PARA cada arquivo
				for (Requirement requirement : requirements)
				{
					String   requirementId  = requirement.getCode();
					String   requirementName  = requirement.getTitle();

					// Cria o tópico
					Topic classTopic;
					Topic requirementTopic = topicMap.createTopic();

					if	 (requirementId.startsWith("RF"))
						classTopic = functionalRequirementTopic;
					else if (requirementId.startsWith("RNF"))
						classTopic = nonFunctionalRequirementTopic;
					else if (requirementId.startsWith("RI"))
						classTopic = inverseRequirementTopic;
					else // TIPO NÃO IDENTIFICADO...
						continue;

					requirementTopic.addType(classTopic);
					requirementTopic.createTopicName(requirementId+"
"+requirementName,null);

		requirementTopic.addSubjectIdentifier(topicMap.createLocator("locator:/baseRequireme
nt#"+requirementId));

		requirementTopic.createOccurrence(requirement.getText(),textOccurrenceTopic,null);

					LOGGER.debug("Created topic - "+requirementId);
				}

				//////////////////////////////////////////////////////////
				// 5. Adiciona o mapa a lista de saida
			//
				//////////////////////////////////////////////////////////
				return topicMap;
			}
			catch(TMAPIException e)
			{
				e.printStackTrace();
				LOGGER.error("Internal error while creating topic map");
				return null;
			}
		}

		public static TopicMap setAssociations(TopicMap topicMap,Map<String,Set<String>>
associationsMap,String theme)
		{
			LOGGER.info("Mapping Requirements Associations...");
```

```
                // Stemmiza tema principal
                theme = new PortugueseStemmer().stem(theme);

                // Recupera subtemas
                Collection<String> subthemes = associationsMap.keySet();

                try
                {
                        ////////////////////////////////////////////////////////
                        // 1. Recupera Mapa        e Índice
                        //
                        ////////////////////////////////////////////////////////

                        TopicsIndex                topicMapIndex        =        (TopicsIndex)
topicMap.getHelperObject(org.tmapi.index.core.TopicsIndex.class);

                        if(topicMapIndex.getFlags().isAutoUpdated()==false)
                                topicMapIndex.reindex();

                        ////////////////////////////////////////////////////////
                        // 2. Inicializa Tópicos Básicos
                //
                        ////////////////////////////////////////////////////////
                        LOGGER.debug("Make basic topics");

                        // Tópico Tema
                        Topic baseThemeTopic = topicMap.createTopic();
                        baseThemeTopic.createTopicName("Tema",null);

                        // Tópico Subtema
                        Topic baseSubthemeTopic = topicMap.createTopic();
                        baseSubthemeTopic.createTopicName("Subtema",null);

                        Topic themeSubthemeAssociationTopic = topicMap.createTopic();

        themeSubthemeAssociationTopic.createTopicName("Relacionamento",null);

                        Topic associationTopic = topicMap.createTopic();
                        associationTopic.createTopicName("Relacionamento",null);

                        Topic associationRoleTopic = topicMap.createTopic();
                        associationRoleTopic.createTopicName("Relacionado A",null);

                        ////////////////////////////////////////////////////////
                        // 4. Cria tópico do tema
                        //
                        ////////////////////////////////////////////////////////

                        Topic themeTopic = topicMap.createTopic();
                        themeTopic.createTopicName(theme,null);
                        themeTopic.addType(baseThemeTopic);

                        ////////////////////////////////////////////////////////
                        // 4. Cria associações tema-subtema e subtema-requisito
                //
                        ////////////////////////////////////////////////////////

                        // PARA cada subtema
                        for (String subtheme : subthemes)
```

```
                {
                        LOGGER.debug("Mounting associations for subtheme: "+subtheme);

                        //
                        // 4.1. tema-subtema
                        //

                        Topic subthemeTopic = topicMap.createTopic();
                        subthemeTopic.createTopicName(theme+"_"+subtheme,null);
                        subthemeTopic.addType(themeTopic);

                        // Recupera requisitos associados
                        Collection<String> associatedRequirements = associationsMap.get(subtheme);

                        // Inicializa requisito
                        Topic requirementTopic = null;

                        // PARA cada requisito
                        for (String requirementId : associatedRequirements)
                        {
                                LOGGER.debug("- "+requirementId);

                                Locator subjectIdentifier = topicMap.createLocator("locator:/baseRequirement#"+requirementId.replaceAll(".txt",""));
                                requirementTopic = topicMapIndex.getTopicBySubjectIdentifier(subjectIdentifier);

                                Association subthemeRequirementAssociation = topicMap.createAssociation();


        subthemeRequirementAssociation.setType(associationTopic);

        subthemeRequirementAssociation.createAssociationRole(subthemeTopic,associationRoleTopic);

        subthemeRequirementAssociation.createAssociationRole(requirementTopic,associationRoleTopic);

                        }
                }

                ////////////////////////////////////////////////////////////
                // 5. Adiciona o mapa a lista de saida
        //
                ////////////////////////////////////////////////////////////
                return topicMap;
        }
        catch(TMAPIException e)
        {
                e.printStackTrace();
                return null;
        }
}

public static String serialize(TopicMap topicMap)
{
        // Cria stream de saída
```

```
                ByteArrayOutputStream xmlOutputStream = new ByteArrayOutputStream();

                try
                {
                        ////////////////////////////////////////////////////////
                        // 1. Serializa o mapa
                                //
                        ////////////////////////////////////////////////////////

                        XTMSerializer xtmS = new XTMSerializer();
                        xtmS.serialize(xmlOutputStream, topicMap);

                        ////////////////////////////////////////////////////////
                        // 2. Adiciona o mapa a lista de saida
                //
                        ////////////////////////////////////////////////////////
                        return xmlOutputStream.toString();
                }
                catch(TMAPIException e)
                {
                        e.printStackTrace();
                        return null;
                }
        }
}
```

Agente Rastreador (tracker.java)

```
package requisitos.agents;

import java.util.List;
import java.util.Map;

import org.midas.as.agent.board.Board;
import org.midas.as.agent.board.Message;
import org.midas.as.agent.board.MessageListener;
import org.midas.as.agent.templates.Agent;
import org.midas.as.agent.templates.LifeCycleException;
import org.midas.as.agent.templates.ServiceException;

public class Tracker extends Agent implements MessageListener
{
        @Override
        protected void lifeCycle() throws LifeCycleException, InterruptedException
        {
                // Se registra no BlackBoard
                Board.addMessageListener("Tracker",this);
        }

        @Override
        public void provide(String service, Map<String, Object> in, List<Object> out) throws
ServiceException
        {

        }

        public void boardChanged(Message msg)
        {

        }
}
```

Agente Validador (validator.java)

```
package requisitos.agents;

import java.util.List;
import java.util.Map;

import org.midas.as.agent.board.Board;
import org.midas.as.agent.board.Message;
import org.midas.as.agent.board.MessageListener;
import org.midas.as.agent.templates.Agent;
import org.midas.as.agent.templates.LifeCycleException;
import org.midas.as.agent.templates.ServiceException;

public class Validator extends Agent implements MessageListener
{
	@Override
	protected void lifeCycle() throws LifeCycleException, InterruptedException
	{
		// Se registra no BlackBoard
		Board.addMessageListener("Validator",this);
	}

	@Override
	public void provide(String service, Map<String, Object> in, List<Object> out) throws
ServiceException
	{

	}

	public void boardChanged(Message msg)
	{

	}
}
```

Agente Verificador (verifier.java)

```
package requisitos.agents;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.text.DecimalFormat;
import java.util.Collections;
import java.util.List;
import java.util.Map;
import java.util.ResourceBundle;
import java.util.Set;

import javax.mail.MessagingException;
import javax.mail.internet.AddressException;

import org.midas.as.AgentServer;
import org.midas.as.agent.board.Board;
import org.midas.as.agent.board.Message;
import org.midas.as.agent.board.MessageListener;
```

```
import org.midas.as.agent.templates.Agent;
import org.midas.as.agent.templates.LifeCycleException;
import org.midas.as.agent.templates.ServiceException;
import org.midas.as.manager.execution.Logger;

import requisitos.business.entities.Requirement;
import requisitos.business.entities.RequirementDocument;
import requisitos.business.entities.User;
import requisitos.business.entities.VerifyValidateProccess;
import requisitos.dao.RequirementDocumentDao;
import requisitos.util.CalculateSimilarity;
import requisitos.util.ValuedAttribute;
import requisitos.web.email.Email;

public class Verifier extends Agent implements MessageListener
{
	private static String className = "[VERIFIER AGENT]  ";

	ResourceBundle resourceBundle = ResourceBundle.getBundle("core");

	@Override
	protected void lifeCycle() throws LifeCycleException, InterruptedException
	{
		// Se registra no BlackBoard
		Board.addMessageListener("Verifier",this);
	}

	@Override
	public void provide(String service, Map<String, Object> in, List<Object> out) throws
ServiceException
	{

	}

	public void boardChanged(Message msg)
	{
		String title = msg.getTitle();

		if ( title.equals("PreVerificationStage") )
		{
			String documentName = msg.getContent();

			// Notifica os outros agentes
			Logger.addEntry(className+"Running similarity tests on
"+documentName,true);
			try
			{
				calculateSimilarity(documentName);
				Logger.addEntry(className+"Similarity tests on
"+documentName+" completed, notifying Communicator Agent",true);

	Board.writeOnBoard(1,"Communicator","SimilarityTest",documentName,this);
			}
			catch (Exception e)
			{
				Logger.addEntry(className+"Error while running similarity
tests on "+documentName,false);
				e.printStackTrace();
			}
```

```
		}
	}

	private void calculateSimilarity(String documentName) throws Exception
	{
		RequirementDocumentDao requirementDao = (RequirementDocumentDao)AgentServer.getApplicationContext().getBean("requirementDocumentDao");

		// 1. Recupera o diretório do perl e do pretex
		String perlPath = resourceBundle.getString("perl.path")+"\\bin";
		String pretexPath = resourceBundle.getString("pretex.path");

		// 2. Limpa os diretórios docs,stembase e discover
		cleanDirectories(pretexPath);
		System.out.println("path pretex "+pretexPath);

		// 3. Cria os arquivos de requisito
		RequirementDocument document = requirementDao.getRequirementDocumentByName(documentName);
		Set<Requirement> requirements = document.getRequirements();
		createRequirementFiles(pretexPath,requirements);

		// 4. Invoca o processo "stem.pl"
		invokeStem(pretexPath);

		// 5. Invoca o processo "report.pl"
		invokeReport(pretexPath);

		// 6. Calcula as similaridades
		List<ValuedAttribute> similarityList = calculateSimilarity();

		// 7. Monta e envia o relatório
		sendEmail(documentName,similarityList);
	}

	private void sendEmail(String documentName, List<ValuedAttribute> similarityList)
	{
		String report = generateReport(similarityList);

		RequirementDocumentDao rdao = (RequirementDocumentDao)AgentServer.getApplicationContext().getBean("requirementDocumentDao");
		List<VerifyValidateProccess> vvproccessList = rdao.getVerifyValidateProccessByDocumentName(documentName);

		VerifyValidateProccess vvproccess = vvproccessList.get(vvproccessList.size()-1);
		Set<User> users = vvproccess.getUsers();

		for (User user : users)
		{
			try
			{
				Email.EnviaEmail(user.getEmail(), "Rq.NET - Relatório de Requisitos Similares em "+vvproccess.getRequirementDocument().getName(),
					"Caro(a) "+user.getUsername()+",\n\n" +
					"Os testes de similaridade em  "+vvproccess.getRequirementDocument().getName()+" foram concluídos.\n\n"+
```

```
                                                                    "O sistema encontrou
"+similarityList.size()+" pares de requisitos similares, listados abaixo, do par mais "+
                                                                    "similar, para o menos
similar: \n\n"+ report);
                        }
                        catch (AddressException e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                        } catch (MessagingException e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                        } catch (Exception e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                        }
                }

        }

        private String generateReport(List<ValuedAttribute> similarityList)
        {
                System.out.println(className+" Gerando Relatório");

                DecimalFormat formatter = new DecimalFormat("0.00");

                String report = "";

                report += "Similaridade\tRequisitos\n";

                Collections.sort(similarityList);
                Collections.reverse(similarityList);

                for (ValuedAttribute<String> va : similarityList)
                {
                        Double factor = va.getAttributeValue();
                        String reqs = va.getAttribute();

                        report += formatter.format(factor)+"\t\t"+reqs+"\n";
                }

                return report;
        }

        private List<ValuedAttribute> calculateSimilarity()
        {
                System.out.println(className+" Calculando Similaridades");
                return new CalculateSimilarity().calculateSimilarity();
        }

        private void invokeReport(String pretexPath) throws IOException,InterruptedException
        {
                System.out.println(className+" Rodando report.pl");

                //Roda "report.pl"
                Runtime runtime = Runtime.getRuntime();
                Process proc = runtime.exec("cmd.exe /c report.bat C:\\Perl\\bin\\ 2> nul: 1>
nul:",new String[]{},new File(pretexPath));

                InputStream inputstream = proc.getInputStream();
                InputStreamReader inputstreamreader = new InputStreamReader(inputstream);
```

```
                BufferedReader    bufferedreader    = new BufferedReader(inputstreamreader);

                String line;

                while ((line = bufferedreader.readLine()) != null)
                {
//                      System.out.println(line);
                }

                proc.waitFor();
        }

        private void invokeStem(String pretexPath) throws IOException,InterruptedException
        {
                System.out.println(className+" Rodando stem.pl");

                // Roda "stem.pl"
                Runtime runtime = Runtime.getRuntime();
                Process proc = runtime.exec("cmd.exe /c stem.bat C:\\Perl\\bin\\ 2> nul: 1>
nul:",new String[]{},new File(pretexPath));

                InputStream          inputstream          = proc.getInputStream();
                InputStreamReader inputstreamreader = new InputStreamReader(inputstream);
                BufferedReader    bufferedreader    = new BufferedReader(inputstreamreader);

                String line;

                while ((line = bufferedreader.readLine()) != null)
                {
//                      System.out.println(line);
                }

                proc.waitFor();
        }

        private void createRequirementFiles(String pretexPath,Set<Requirement> requirements)
throws IOException
        {
                System.out.println(className+" Criando arquivos de requisitos");

                for (Requirement requirement : requirements)
                {
                        File                reqFile                =                new
File(pretexPath+"//docs//"+requirement.getCode()+".txt");
                        BufferedWriter writer  = new BufferedWriter(new FileWriter(reqFile));

                        writer.append(requirement.getCode()+"
"+requirement.getTitle()+"\n"+requirement.getText());
                        writer.close();
                }
        }

        private void cleanDirectories(String pretexPath)
        {
                System.out.println(className+" Limpando Diretórios");

                File docsDir = new File(pretexPath+"//docs");
                File[] docsFiles = docsDir.listFiles();

                for (File file : docsFiles)
```

```
                {
                        file.delete();
                }

                File stembaseDir = new File(pretexPath+"//stembase");
                File[] stembaseFiles = docsDir.listFiles();

                for (File file : docsFiles)
                {
                        file.delete();
                }

                File discoverDir = new File(pretexPath+"//discover");
                File[] discoverFiles = docsDir.listFiles();
                System.out.println(className+"discoverDir "+discoverDir);
                for (File file : docsFiles)
                {
                        file.delete();
                }
        }
}
```

Anexo A
Stopwords utilizadas

a	à	abaixo	acaso	acerca
acima	acola	acolá	ademais	adentro
adiante	afinal	afora	agora	agorinha
ai	aí	ainda	alem	além
algo	alguem	alguém	algum	alguma
algumas	alguns	ali	alias	aliás
amiude	amiúde	ante	antes	ao
aonde	aos	apenas	apesar	apos
após	apud	aquela	àquela	aquelas
àquelas	aquele	àquele	aqueles	àqueles
aqui	aquilo	àquilo	as	às
assim	ate	até	atras	atrás
atraves	através	basicamente	bastante	bastantes
bem	bom	ca	cá	cada
cade	cadê	cadern	caso	cem
certa	certamente	certas	certeiramente	certo
certos	chez	cinco	cinquenta	com
comigo	como	comumente	conforme	confronte
conosco	conquanto	consequentemente	conseqüentemente	consigo
consoante	contanto	contigo	contra	contudo
convosco	cuja	cujas	cujo	cujos
da	dai	daí	dali	dantes
daquela	daquelas	daquele	daqueles	daqui
daquilo	das	de	debaixo	defronte
dela	delas	dele	deles	demais
dentre	dentro	depois	desde	dessa
dessas	desse	desses	desta	destas
deste	destes	detras	detrás	deveras
dez	dezenove	dezesseis	dezessete	dezoito
diante	disso	disto	diversos	do
dois	donde	doravante	dos	doze
duas	dum	duma	dumas	duns
durante	duzentos	e	é	eis
ela	elas	ele	eles	em
embaixo	embora	enfim	enquanto	entanto
entao	então	entre	entretanto	essa
essas	esse	esses	esta	estas
este	estes	eu	exatamente	exceto
exceto	felizmente	fora	frequentemente	freqüentemente
graças	hoje	ibidem	idem	in
inclusive	inda	infelizmente	inicialmente	isso
isto	ja	já	jamais	la
lá	largamente	lha	lhas	lhe
lhes	lho	lhos	logo	mais
mal	malgrado	mas	me	mediante
melhor	menos	meramente	mesma	mesmas
mesmo	mesmos	meu	meus	mil
milhao	mim	minha	minhas	mui

muita	muitas	muitissimo	muitíssimo	muito
muitos	mutuamente	na	nada	nadinha
nalgum	nalguma	nalgumas	nalguns	nao
não	naquela	naquelas	naquele	naqueles
naquilo	nas	nela	nelas	nele
neles	nem	nenhum	nenhuma	nessa
nessas	nesse	nesses	nesta	nestas
neste	nestes	ninguem	ninguém	nisso
nisto	no	nos	nós	nossa
nossas	nosso	nossos	noutra	noutras
noutro	noutros	novamente	nove	novecentos
noventa	num	numa	numas	nunca
nunquinha	nuns	o	oitenta	oito
oitocentos	onde	ontem	onze	ora
os	ou	outra	outras	outrem
outro	outrora	outros	outrossim	para
pela	pelas	pelo	pelos	per
perante	pero	pois	por	porem
porém	porquanto	porque	porquê	portanto
porventura	possivelmente	posteriormente	posto	pouca
poucas	pouco	poucos	pra	praquela
praquelas	praquele	praqueles	praquilo	pras
praticamente	prela	prelas	prele	preles
preste	prestes	previamente	primeiramente	principalmente
priori	pro	pró	pronto	propria
própria	proprias	próprias	proprio	próprio
proprios	próprios	pros	prós	proximo
próximo	quais	quaisquer	qual	qualquer
quando	quanta	quantas	quanto	quantos
quao	quão	quarenta	quase	quatorze
quatro	quatrocentos	que	quê	quem
quer	quiça	quiçá	quinhentos	quinze
raramente	realmente	recentemente	salvante	salvo
se	segundo	seguramente	seis	seiscentos
seja	sem	sempre	senao	senão
sequer	sessenta	sete	setecentos	setenta
seu	seus	sim	simplesmente	so
só	sob	sobre	sobremaneira	sobremodo
sobretudo	somente	sua	suas	tais
tal	talvez	tambem	também	tampouco
tanta	tantas	tanto	tantos	tao
tão	tao-so	tão-só	tao-somente	tão-somente
te	teu	teus	ti	tirante
toda	todas	todavia	todo	todos
tras	trás	tres	treze	trezentos
trilhao	trinta	tu	tua	tuas
tudo	um	um	uma	umas
uns	varias	várias	varios	vários
versus	vezes	via	vice-versa	vinte
visto	voce	você	voces	vocês
vos	vós	vossa	vossos	vulgo