Ambientes Complexos: representando conceitos, estruturas geoespaciais, suas dinâmicas e relacionamentos de forma computável

Pier-Giovanni Taranti (1), Marcelo Corsino Ferreira (1) e Ricardo Choren (2)
(1) Centro de Análises de Sistemas Navais, Praça Barão de Ladário s/n, Rio de Janeiro/RJ
(2) Instituto Militar de Engenharia, Pça General Tibúrcio 80, Rio de Janeiro/RJ

***Resumo*—Sistemas computacionais utilizados na área de defesa, tais como sistemas de comando e controle, treinadores táticos e sistemas de jogos de guerra, utilizam-se de alguma abstração do ambiente real, representada internamente no sistema em um modelo computável. Devido ao domínio dos dados tratados, as características e relações físicas e conceituais existentes entre as entidades representadas são importantes, bem como a sua variação no tempo. Este artigo apresenta a arquitetura de um modelo computacional que representa os conceitos do ambiente em uma ontologia e os processos em uma plataforma que provê suporte a georeferenciamento do contexto espacial e verificação de informações válidas em uma situação.**

***Palavras-chaves* – ambientes complexos, representação de ambientes, sistemas multiagentes**

I. INTRODUÇÃO

Sistemas computacionais como sistemas de comando e controle, treinadores táticos e sistemas de jogos de guerra são sistemas que possuem aplicação na área militar, policial, defesa civil e controle de meios móveis, por exemplo. Estes sistemas possuem como característica comum o uso de uma abstração do ambiente real para prover dados aos operadores do sistema. É importante destacar que as decisões tomadas pelo sistema e seus operadores tratam risco sobre vidas e material.

O ambiente real é composto por elementos físicos de dimensões e características com ordens de grandeza diversa, que podem ou não se sobrepor. Estes elementos possuem qualidades e relacionamentos que se alteram conforme a situação em que se encontrem, sendo a situação de um elemento formada pela composição de todos os contextos que podem afetá-lo, ou seja, pelo conjunto de informações que são importantes no momento [1].

Este ambiente é complexo, sendo composto por contextos sociais e espaciais, atores e relacionamentos existentes. Pequenas perturbações em nível micro podem se propagar em nível macro, afetando o ambiente como um todo, porém as conseqüências são imprevisíveis, por depender também de propriedades que podem variar ao longo do tempo [2].

Em Sistemas Multiagentes (SMA), o ambiente é um elemento de primeira ordem [3] que fornece abstração para a criação de um mundo virtual para os agentes e pode ou não representar contextos sociais (p. ex. organizações) e físicos (p.e., objetos, áreas) do mundo real, bem como a forma como estes contextos e os agentes se relacionam, i.e., estabelecer regras do domínio considerado.

Existem diversas abordagens para a representação de ambientes em SMA: [4] propõe uma representação do ambiente em três camadas, separando os modelos da realidade, conceitual e mental; [5] apresenta uma ontologia para definição de ambientes que utiliza a ELMS [6] e contempla contextos espaciais. Existem também trabalhos sobre regulação de SMA: [7] apresenta o DynaCROM, que define uma ontologia que relaciona os contextos dos agentes a estrutura normativa do SMA e cujo foco é provimento de normas contextuais aos agentes; [8] apresenta o XMLaw, cuja abordagem é centrada no contexto de comunicações dos agentes.

Tanto as abordagens para a representação de ambientes quanto as abordagens para regulação de SMA não permitem uma representação espacial contínua, não oferecendo suporte a sistemas de comando e controle, simulações geográficas e outros. Para SMAs que possuem contextos geográficos, não se pode desvincular a interação entre agentes e a verificação de informações sensíveis ao georeferenciamento. Esta visão considera características de sistemas com ambientes dinâmicos e georeferenciados, nos quais as informações aplicáveis a um agente podem variar no tempo e conforme suas coordenadas no espaço virtual (i.e., contexto espacial) ou de acordo com o papel que desempenhe e as organizações a que esteja filiado (i.e., contexto social).

Este artigo apresenta uma arquitetura que possibilita representar ambientes georeferenciados e dinâmicos de SMA em um modelo computacional que associa mecanismos para verificação de normas e informações contextuais. A arquitetura, chamada Dominium, é composta por uma ontologia que representa o domínio do sistema e por uma plataforma que tem por fim informar os agentes a quais normas eles estão sujeitos, verificar se os mesmos as estão cumprindo e executar medidas previstas para violações.

Este artigo está estruturado da seguinte forma: A seção 2 apresenta os conceitos de ambiente e contexto. A seção 3 apresenta a ontologia de domínio e a arquitetura que compõem a solução. Na seção 4 é apresentado um exemplo que ilustra o funcionamento da solução. A seção 5 apresenta trabalhos relacionados e a conclusão é apresentada na seção 6.

II. Ambientes e seus Contextos em um SMA

Embora não haja uma definição amplamente aceita de ambiente para SMA [3], este trabalho considera ambiente como sendo a parte do SMA que é externa aos agentes e que é capaz de prover informações que podem ser usadas pelos agentes em seus processos decisórios. Pode-se abstrair o ambiente em um SMA como sendo um mundo virtual, composto de estruturas sociais e espaciais e regras que regulam o uso destes recursos ou as interações ocorridas sob suas influencias (i.e., ações dos agentes). Um exemplo seria a representação virtual de um aeroporto, onde os aviões são agentes subordinados a companhias aéreas (organização social) e devem cumprir corredores de aproximação ou manter-se em níveis de espera (estruturas espaciais) de acordo com as normas de operação de aeródromo.

Este mundo virtual é composto pelo conjunto de contextos sociais e espaciais [9] do SMA, onde contexto é qualquer informação que possa ser usada para caracterizar a situação de um agente [1]. É importante que agentes que se encontrem na mesma situação, i.e., mesmo conjunto de contextos, sofram a mesma influência do seu mundo virtual. Isso implica que os conceitos deste ambiente devem ser compartilhados pelos agentes nele situados. Os contextos sociais e espaciais podem ser dinâmicos ou estáticos, conforme variem ou não seus estados no decorrer do tempo.

Para especificar e relacionar os conceitos relativos aos contextos (espacial e social) e às normas definiu-se uma ontologia para descrever o ambiente de um SMA. A ontologia de ambiente apresentada reflete conceitos do nível de mediação de interações [9], e permite representar todos contextos de um sistema georeferenciado em espaço contínuo, e as associações que esses conceitos possuem em seu mundo virtual. É importante ressaltar que nesta ontologia não existe associação entre contextos sociais e espaciais, ou seja, uma organização não é localizada geograficamente, mas sim os agentes que lhe são subordinados. Outro detalhe importante é que não há correlação entre os contextos dos agentes e o *host* no qual o agente é executado. Com isto, possibilitam-se contextos espaciais móveis, organizações que permeiam vários contextos espaciais e papéis que são executados em várias organizações.

Na ontologia, os papéis (*Role*) que um agente pode desempenhar e as organizações (*Organization*) existentes são contextos sociais (*SocialContext*). Contextos espaciais (*SpatialContext*) são regiões (*Region*) ou objetos geográficos (*GeographicObject*) do espaço virtual, que podem ser georeferenciados e que afetam o comportamento de agentes devido a propriedades espaciais (e.g distancia, dentro de). O conjunto total dos contextos forma o ambiente (*Environment*). Uma situação é composta por contextos, possuido um conjunto de dados (*SituationData*) formado por informações (*Information*) e normas (*Norm*), onde as normas regulam ações (*Action*) que podem ser executadas no ambiente.

A ontologia desenvolvida fornece recursos para a representação do ambiente. Contudo, para executar as tarefas atinentes ao ambiente (regular o acesso a recursos compartilhados do ambiente e para mediar as interações entre agentes), e ainda prover o suporte ao georeferenciamento do ambiente espacial, existe necessidade de processamento destas informações. A próxima seção apresenta uma arquitetura que tem por fim suprir estas necessidades.

III. Dados e Processos do Ambiente em Uma Arquitetura de Software

Este trabalho está focado no suporte a sistemas com ambientes de espaço virtual contínuo e georeferenciado. Para isso, além da ontologia apresentada anteriormente, foi desenvolvida uma arquitetura de modelo dinâmico do domínio, capaz de representar agentes, ambiente e estrutura normativa com todos os seus processos. Essa arquitetura possui como componentes uma plataforma composta por um conjunto de agentes, responsáveis pela dinâmica e pelas tarefas do ambiente, um banco de dados geográficos e bibliotecas de verificações de normas e de conseqüências de violações, cujas classes são individualizadas para cada situação e por uma ontologia de domínio. A ontologia de domínio é uma extensão da ontologia de ambiente, que representa também conceitos e relações dos agentes e do conjunto de informações e normas que vigora no sistema.

Este modelo encerra conhecimento sobre o ambiente de forma processável, permitindo reduzir o esforço na construção do sistema principal. A arquitetura favorece o reuso dos componentes, especificamente da ontologia do domínio. Esta facilidade é favorável em se considerando sistemas de simulação, inclusive os georeferenciados, que podem se valer de mundos virtuais para realizar experimentos de simulação baseada em SMAs. As subseções seguintes descrevem com maiores detalhes as partes componentes da solução.

A. A Ontologia de Domínio

A ontologia do domínio utilizada pela arquitetura, apresentada na fig. 1, é uma extensão da ontologia do ambiente. A ontologia original foi acrescida dos conceitos descritos abaixo, e foram estabelecidos relacionamentos que permitem obter a situação de um agente através de inferências e consultas padronizadas pelos agentes da arquitetura.

O conceito *Agent* representa os agentes normativos (i.e., considerados na ontologia e que são sensíveis a normas e contextos) do sistema, e possui as propriedades *isIn*, *play* e *subordinatedOf* associando-o aos conceitos *Region*, *Role* e *Organization*, respectivamente.

Foram inseridas subclasses ao conceito de *Norm*: *Obligation*, *Permition* e *Prohibition*, seguindo a lógica deontica. Os conceitos *Verify* e *Consequence* são utilizados para realizar verificação de transgressão de normas. *Norm* associa-se ao conceito de *Verify* pela propriedade *execute*, e *Verify* possui a propriedade *canCause* associando-o a *Consequence*. A associação entre as instancias de *Norm* e *Consequence* na ontologia e no SMA dá-se pelo nome, que corresponde ao nome de uma classe capaz de executar as ações requeridas.

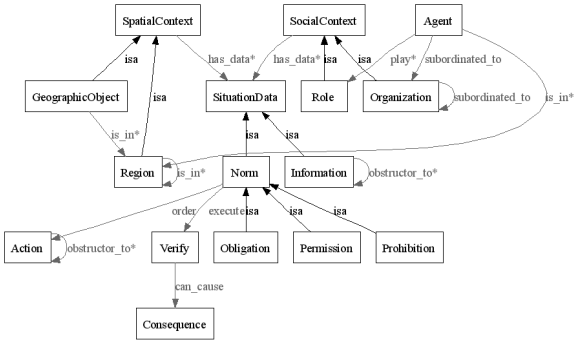

Figura 1. Ontologia de Domínio

B. A Plataforma do Dominium

A plataforma é composta por um conjunto de agentes responsáveis por executar as tarefas relativas à regulação (i.e., a verificação das normas e suas conseqüências) , um banco de dados de informações geográficas (GisDB), que fornece suporte a questão de georeferenciamento, e pelas bibliotecas de verificação de normas e conseqüências de violações. O Diagrama de Agentes [10] que contém a arquitetura da plataforma é apresentado na fig. 2.

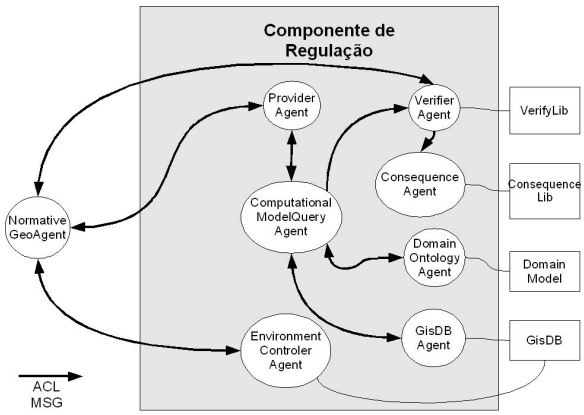

Figura 2. Arquitetura do *Dominium*

1) Os Agentes do Dominium: Para construção dos agentes normativos da aplicação são utilizados modelos, para as quais o analista deverá definir o conjunto de variáveis que representam o estado do agente normativo no domínio da aplicação. Esse estado será utilizado pelo *Dominium* para verificação das informações e do cumprimento das normas pelos agentes da aplicação. Estes modelos são representados por *NormativeAgent* e *NormativeGeoAgent*.

Para suportar a dinâmica do ambiente e a verificação de normas, é fornecido um conjunto de sete agentes: *ProviderAgent*, *GisDBAgent*, *DomainOntologyAgent*, *ComputacionalModelQueryAgent*, *VerifierAgent*, *ConsequenceAgent* e o *EnvironmentControlerAgent*, que interagem por troca de mensagens ACL [11].

Nesta arquitetura o *ProviderAgent* recebe requisições de informação e normas dos agentes normativos, e encaminha o pedido ao *ComputacionalModelQueryAgent*. Este recebe esta requisição, incluindo-a na sua lista de tarefas que contém uma agenda de verificação contínua de todos os Agentes Normativos do sistema. Para executar suas tarefas o *ComputacionalModelQueryAgent* encaminha requisição de contextos geográficos ao *GisDBAgent* e de inferência sobre dados da ontologia ao *DomainOntologyAgent*, recebendo deste último uma lista de normas aplicável ao agente pesquisado. Esses dados são então encaminhados ao *ProviderAgent* e ao *VerifierAgent*, para verificação do cumprimento de normas. O *ProviderAgent*, ao receber estes dados, transmite-os como resposta a requisição do *NormativeAgent*.

O *GisDBAgent* e o *DomainOntologyAgent* realizam a interface entre os demais agentes da plataforma com a ontologia do domínio e o GisDB, respectivamente. O *GisDBAgent* é responsável por converter as coordenadas do espaço virtual do sistema para o padrão SFS-SQL do OpenGIS. O acoplamento do *Dominium* ao banco de dados é fraco [12] e dinâmico [13]. Este agente recebe requisições sobre uma coordenada (ex. dentro de, distante de) e responde com uma lista de contextos geográficos que atendam a relação informada. Seu uso mais freqüente é informar quais contextos geográficos em que um determinado agente está inserido (i.e., regiões). O *DomainOntologyAgent* responde solicitações sobre a ontologia, realizando inferência sobre a mesma de acordo com as regras da *Web Ontology Language* para relacionamentos em Lógica Descritiva (OWL-DL). Para realizar esta tarefa utiliza mecanismo de inferência OWL-DL. O DomainOntologyAgent utiliza-se das propriedades existentes na ontologia para responder a quais normas o agente está sujeito.

O *VerifierAgent* recebe como informação o nome do agente e uma lista de normas aplicáveis. Cabe a este agente verificar se todas as normas estão sendo cumpridas. Para cada norma transgredida ele gera uma informação ao agente transgressor e ao *ConsequenceAgent*. O *ConsequenceAgent* recebe a norma e o transgressor como informação e executa os procedimentos previstos no sistema para esta ocorrência, que podem afetar ou não o agente normativo, sendo isto uma decisão de implementação.

A consistência do GisDB e da Ontologia são responsabilidade do *EnvironmentControllerAgent* (ECA). Sua tarefa é manter os contextos geográficos com suas coordenadas atualizadas no tempo, permitindo assim a representação de processo no espaço. Para isto, cada contexto geográfico deverá possuir um planejamento cinemático no GisDB para uso do ECA. Quando a existência de um contexto espacial terminar ou iniciar, cabe ao ECA iniciar as ações para retirá-los ou inseri-los na ontologia.

2) Georeferenciamento: Adicionando propriedades geográficas: O suporte georeferenciamento é muito mais amplo que um simples sistemas de coordenadas cartesianas associado a agentes e contextos geográficos. Ele implica na solução de consultas e problemas como verificação de adjacências, deslocamento de regiões e conversão de objetos geográficos, entre outros. Essas propriedades permitem um tratamento computacional a representações espaciais de objetos.

Para possibilitar o suporte ao georeferenciamento optou-se por acoplar o SMA a um GisDB capaz de responder a consultas padronizadas SFS-SQL e aderente a padrões OpenGIS. Com isso, o tratamento das informações geográficas é realizado de forma transparente ao SMA.

O GisDB do Dominium é instanciado em um servidor de Banco de Dados com extensão geográfica e seu esquema possui cinco tabelas fixas, sendo três para registro dos contextos (i.e., áreas e objetos geográficos), agentes e suas posições no tempo e as outras duas para registro do planejamento de deslocamento das áreas e objetos, incluindo ETA e ETD dos pontos de suas rotas. A inserção de dados pode ser realizada por aplicações externas.

3) As Bibliotecas de Verificações e Conseqüências: A arquitetura prevê que sejam desenvolvidas bibliotecas de verificação de normas e conseqüências para o domínio da aplicação. As classes de verificação devem utilizar dados do estado dos agentes normativos para verificar a consonância com as normas e as classes de conseqüências devem prever ações sobre o ambiente e agentes caso uma violação se verifique. O acionamento das classes destas bibliotecas é realizado pelos *VerifierAgent* e *ConsequenceAgent* de forma dinâmica, conforme informações inferidas sobre a ontologia.

IV. Exemplo de Aplicação da Arquitetura

Um Sistema de Informações de Monitoramento de Tráfego de Navios (VTMI, em inglês) pode ser definido como um sistema de simulação visual interativa [14], onde cada navio possui representação no sistema mantida por um agente que é georeferenciado em coordenadas e atualizado periodicamente com dados enviados pelo navio real (p.e., rumo, velocidade, posição). Este tipo de sistema apóia diversas atividades, como controle do mar territorial e serviços de busca e salvamento (SAR). Abaixo é hipoteticamente descrita a aplicação da arquitetura do Dominium ao VTMI e de um cenário do mesmo.

A. Instanciando a Ontologia do Domínio

O desenvolvedor deverá inicialmente identificar quais conceitos da ontologia de domínio devem ser estendidos em subclasses. Argumentos que justifiquem a especialização ou não do contexto são decisão do desenvolvedor. No caso do VTMI, os contextos que sofrem especialização são *Role* e *Region*: *Role* foi estendido em uma taxonomia, na qual os papéis são agrupados em uma hierarquia de classes (p.e., navio-mercante -> navio-tanque -> navio-químico de transporte de gases liquefeitos). Neste domínio, o papel que um navio exerce (p.e., navio-tanque) altera o conjunto de regras a que ele está sujeito. Também se especializou o conceito *Region* de forma a apresentar abstrações aceitáveis aos Serviços de Busca e Salvamento, de Segurança a Navegação e de Meteorologia (p.e., conceitos para áreas SAR e de mau tempo)

A instanciação da ontologia é realizada através de um editor de ontologias (p.e., Software Protégé [15]), criando-se as subclasses dos conceitos originais da ontologia de domínio. A partir da ontologia estendida, podem-se criar as instâncias que compõem o ambiente e suas normas associadas.

B. Os Agentes Normativos do Sistema

No VTMI, os agentes devem conhecer a posição do navio que representam e as normas que são aplicáveis a estes navios no mundo real, representado em modelo pelo Dominium. Não cabe a esses agentes evitar que os navios reais transgridam regras, mas sim representá-los no sistema e auxiliar o operador humano em suas tarefas. Para que estes agentes recebam as informações de violação de regras, é necessário que o analista defina qual conjunto de variáveis do agente define seu estado no ambiente. O agente responderá uma *request* em ACL, realizada pelo *VerifierAgent*, sobre seu estado com estes dados. Para o VTMI, é necessário utilizar o modelo *NormativeGeoAgent*, que habilita o suporte a georeferenciamento, sendo o estado dos agentes definido pela sua posição em coordenadas, rumo e velocidade no espaço virtual.

C. Verificando Normas e Informações no VTMI

Depois de encerrada a modelagem da ontologia do domínio, o analista deve criar instâncias do conceito Norm na ontologia, prevendo seu uso futuro. Vamos supor que o VTMI seja usado para planejar e coordenar ações de Busca e Salvamento no Mar (Operação SAR). Para isto, é preciso criar uma norma ProhibitionToNavigation, que define áreas nas quais não se pode navegar. Para que seja possível verificar o cumprimento desta norma e definir ações em caso de sua violação, também se faz necessário desenvolver instâncias dos conceitos de Verify e Consequence para esta norma.

Um exemplo de ações que podem ocorrer em uma situação real seria: o Navio Pesqueiro Atum desatraca do porto de Niterói para efetuar pesca na região de Cabo Frio. Antes de partir, entrega para a autoridade portuária um documento chamado Parte de Saída, contendo sua rota prevista, com ETA e ETD em cada ponto de guinada, entre outros dados. O procedimento descrito é cumprido por todos os navios que desatracam de portos.

A autoridade portuária inclui estes dados na ontologia e na base de dados de informações geográficas do VTMI, que cria o *NormativeGeoAgent01* para representar o NPe Atum no sistema, o qual não está contrariando nenhuma norma do sistema e está dentro do contexto espacial *AreaDeControleLeste*, do tipo Region.

Em um momento posterior a saída do Atum, o Serviço de Informações Meteorológicas inclui na ontologia o contexto *RoughSea25*, da classe *RoughSea*. O contexto criado é então associado ao conceito *ProhibitionToNavigation* existente (fig. 3) pela propriedade *hasData*. É criada uma tupla na *CinematicRegionTable* do GisDB onde constam rumo, velocidade e tempo de duração desta área. Também é criada uma tupla na *regionTable*, com as coordenadas que limitam o polígono de *RoughSea25*. Essas representações no GisDB e na ontologia possuem informações relacionadas ao mesmo contexto espacial, que é georeferenciado e possui dinâmica própria, executada pelo *EnvironmentControllerAgent*.

Quando o *NormativeGeoAgente01* solicita suas normas a plataforma *Dominium*, é informado estar violando *ProhibitionToNavigation*, pois se encontra na área *RoughSea25*.

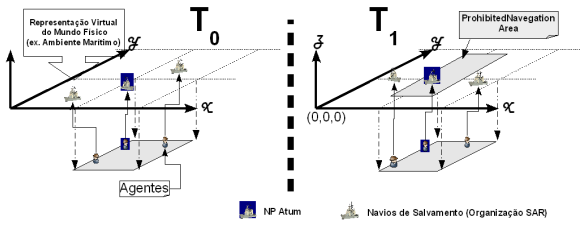

Figura 3. Área Proibida a Navegação no Espaço Virtual Georeferenciado

Ao receber esses dados o agente cumpre sua programação, alterando a cor de sua representação na tela do operador. O Dominium, ao verificar a violação da norma executa as ações de conseqüência previstas para o fato: verifica a embarcação de salvamento mais próxima do NPe Atum e emite ordem de prestar socorro como responsável pelas ações no mar, entre outras ações junto ao serviço SAR (Search and Rescue).

O evento narrado é hipotético, porém ilustra as vantagens do uso do *Dominium* neste caso, onde o uso do modelo dinâmico do domínio associado à estrutura normativa permite auxiliar o planejamento e coordenação das atividades SAR.

V. Trabalhos Relacionados

A arquitetura apresentada trata como um problema único a representação do ambiente e a verificação de normas. O ponto inicial utilizado para o desenvolvimento do Dominium foi o trabalho do DynaCROM [7], com a sua ontologia normativa. O *Dominium*, da mesma forma que o DynaCROM, cria um modelo do domínio em uma ontologia. No entanto, o DynaCROM tem foco na informação de dados contextuais para SMAs, além de não levar em conta, de forma direta, o uso de ambientes georeferenciados e dinâmicos (ambientes cujos contextos mudam de posição ao longo do tempo).

Em [16] é apresentado o relacionamento entre normas e contextos geográficos, sendo este trabalho seqüência de [6] [5], cujo foco é em ambientes para simulações multiagentes. Nestes trabalhos não há suporte a ambientes de espaço contínuo. [4] apresenta uma representação do ambiente em três camadas que representam modelos da realidade, conceitual e mental, visando uso em agentes cognitivos. O trabalho não explicita suporte a localização geográfica. Nenhum dos trabalhos citados possui suporte a verificação do cumprimento das normas ou mecanismos para aplicação de conseqüências em caso de violação. Em [17] é apresentado um trabalho em andamento que prevê o uso de uma plataforma de agentes especificamente para sistemas de comando e controle.

VI. Conclusão

Considerando ambientes dinâmicos e georeferenciados, definiu-se uma arquitetura que objetiva representá-los em um modelo computacional. A arquitetura possui uma ontologia de domínio e uma plataforma responsável pelos processos e verificação de informações e normas. A ontologia permite representar conceitos do ambiente e seus relacionamentos em forma passível de processamento e permite a verificação e alteração destes dados em tempo de execução. A plataforma permite o uso de georeferenciamento apoiada em um GisDB, realizando tarefas de verificação de normas e conseqüências de violações. A arquitetura permite construir modelos dinâmicos de ambiente fracamente acoplados a sistemas de comando e controle e simuladores táticos, por exemplo.

É necessário realizar estudos específicos sobre desempenho, pois nos experimentos realizados, o custo computacional mostrou-se elevado. Estes estudos devem ser centrados em ajustes na freqüência de execução de eventos dos agentes e na modelagem da ontologia.

Referências

[1] A. Dey, "Understanding and using context," *Personal and Ubiquitous Computing*, vol. 5, no. 1, pp. 4–7, 2001.

[2] J. Moffat, *Complexity theory and network centric warfare*, ser. Information age transformation series, D. S. Alberts, Ed. DoD Command and Control Research Program, 2003.

[3] D. Weyns, H. Parunak, F. Michel, T. Holvoet, and J. Ferber, "Environments for Multiagent Systems State-of-the-Art and Research Challenges," in *Proceedings of First International Workshop on Environments for Multiagent Systems (LNCS 3374)*, 2005, pp. 1–47.

[4] P. Chang, K. Chen, Y. Chien, E. Kao, and V. Soo, "From Reality to Mind: A Cognitive Middle Layer of Environment Concepts for Believable Agents," in *Proceedings of First International Workshop on Environments for Multi-Agent Systems (LNCS 3374)*, ser. LNCS 3374, 2005, pp. 57–73.

[5] F. Y. Okuyama, R. H. Bordini, and A. C. Rocha Costa, "An Ontology for Defining Environments within Multi-Agent Simulations," in *Proceedings of First Workshop on Ontologies and Metamodels in Software Engineering (WOMSDE) at SBES'06*, 2006, pp. 1–10.

[6] ——, "ELMS: An Environment Description Language for Multi-Agent Simulations," in *Proceedings of Environments for Multi-Agent Systems (LNCS 3374)*, 2005, pp. 91–108.

[7] C. Felicissimo, C. Lucena, J.-P. Briot, and R. Choren, "Regulating open multi-agent systems with dynacrom," in *Proceedings of 2nd Workshop on Software Engineering for Agent-oriented Systems (SEAS) at SBES'06*, 2006, pp. 95–106.

[8] G. Carvalho, H. Almeida, M. Gatti, G. Ferreira, R. Paes, A. Perkusich, and C. Lucena, "Dynamic Law Evolution in Governance Mechanisms for Open Multi-Agent Systems," in *Proceedings of 2nd Workshop on Software Engineering for Agent-oriented Systems (SEAS) at SBES'06*, 2006, pp. 83–94.

[9] D. Weyns, A. Omicini, and J. Odell, "Environment, First-Order Abstraction in Multiagent Systems," *Journal of Autonomous Agents and Multi-Agent Systems*, vol. 14, no. 1, pp. 5–30, 2007.

[10] M. Nikraz, G. Caire, and P. A. Bahria, "A methodology for the analysis and design of multi-agent systems using jade," *International Journal of Computer Systems Science and Engineering*, vol. 21, no. 2, 2006.

[11] FIPA, "Foundation for intelligent physical agents," http://www.fipa.org/, 2008.

[12] D. Brown, R. Riolo, D. Robinson, M. North, and W. Rand, "Spatial process and data models: toward integration of agent-based models and gis," *Geographical Systems*, vol. 7(1), pp. 25–47, 2005.

[13] A. Gonçalves, A. Rodrigues, and L. Correia, "Multi-Agent Simulation within Geographic Information Systems," in *Proceedings of the 5th International Workshop on Agent-Based Simulation (ABS-2004)*, 2004, pp. 107–112.

[14] P. Bell and R. O'Keefe, "Visual interactive simulation history, recent developments, and major issues," *Simulation*, vol. 49, no. 3, pp. 109–116, 1987.

[15] Protege, "The protege ontology editor and knowledge acquisition system," http://protege.stanford.edu/, 2008.

[16] F. Okuyama, R. Bordini, and A. Rocha Costa, "Augmenting Multi-Agent Environment Descriptions with a Normative Infaestructure," in *Anais do Encontro Nacional de Inteligencia Artificial (ENIA) no SBC'07*, 2007, pp. 1391–1400.

[17] A. Uruguay, P. da Costa, N. Lessa, and C. dos Santos, "C2oliseu: A meta-model for research and development of complex network centric operations," in *13 International Command and Control Research and Technology Symposia (ICCRTS 2008)*, 2008.