# Diretrizes de Mapeamento entre as Fases de Requisitos e Modelagem de Sistemas Multiagente Normativos

Thiago Gomes de Souza, Henrique Rabelo da Silva e Emmanuel Sávio Silva Freire

***Resumo*—O desenvolvimento de sistemas complexos tem sido abordado por meio de sistemas multiagente normativos. Assim, a Engenharia de *Software* orientada a Agente tem proposto técnicas, processos e linguagens de programação e modelagem para apoiar o desenvolvimento desses sistemas. Neste sentido, o *framework* i* normativo e a linguagem NorMAS-ML foram definidos para dar suporte às fases de requisitos e modelagem, respectivamente. O presente artigo teve como objetivo propor um conjunto de diretrizes para possibilitar o mapeamento entre as entidades existentes em i* normativo e NorMAS-ML. Para ilustrar esse mapeamento, dois exemplos de modelagem foram apresentados.**

***Palavras-chave*—*sistemas multiagente normativos, NorMAS-ML, i* normativo, diretrizes de mapeamento.***

## I. Introdução

Os sistemas multiagente normativos (SMAN) são formados por um conjunto de agentes inteligentes que interagem entre si para alcançar os seus objetivos, verificando as normas definidas nesses sistemas para identificar quais as ações que são permitidas, obrigadas ou proibidas de serem executadas [1] [2]. Assim, linguagens de modelagem e de implementação foram propostas para auxiliar o desenvolvimento desses sistemas. Dentre elas, destaca-se a linguagem *Normative Multiagent System Modeling Language* (NorMAS-ML) [3] que possibilita a modelagem das entidades encontradas nesses sistemas junto com as normas. Contudo, essa linguagem não dá suporte a fase de levantamento e análise de requisitos.

No processo de desenvolvimento de *software*, o levantamento de requisitos compreensíveis por todas as partes envolvidas no desenvolvimento do sistema (clientes, analistas, desenvolvedores etc.) é um fator básico e extremamente importante, evitando falhas no entendimento do problema a ser solucionado [4]. Assim, a fase de levantamento e análise de requisitos propicia um melhor entendimento das reais necessidades dos usuários do sistema [5]. Logo, é possível detalhar e priorizar os requisitos permitindo que aqueles essenciais ao sistema possam ser implementados inicialmente. Consequentemente, o usuário tem uma maior probabilidade de receber um sistema que supra as suas necessidades.

Neste sentido, o *framework* i* (*iStar*) [6] foi proposto para auxiliar a fase de requisitos por meio de um modelo conceitual, permitindo que os analistas de requisitos possam modelar os *stakeholders* como atores e suas intenções como objetivos. Assim, consegue-se obter uma compreensão sobre os relacionamentos da organização (que define os requisitos, ou seja, uma empresa) e sobre as razões envolvidas nos processos de decisão [6]. Entretanto, i* não permitia a modelagem de requisitos organizacionais para os SMAN, pois não abordava os elementos normativos nem a interação deles com as entidades de um SMAN. Por isso, Siena et al. [7] propuseram a extensão de i*, chamada de i* normativo, para permitir a modelagem de aspectos normativos, indicando as entidades que são reguladas pelas normas.

Considerando que i* normativo permite a modelagem de requisitos organizacionais para SMAN e NorMAS-ML permite a modelagem das entidades típicas encontradas nesses sistemas, este artigo teve como objetivo propor um conjunto de diretrizes para possibilitar o mapeamento entre as entidades existentes em i* normativo e NorMAS-ML. Assim, espera-se identificar quais as semelhanças e as diferenças conceituais existentes entre eles e possibilitar a integração entre a fase de levantamento e análise de requisitos e a fase de modelagem para o desenvolvimento de SMAN, proporcionando uma visão mais completa do sistema. Vale ressaltar que a relevância dessa integração foi discutida em um trabalho inicial [8].

Este artigo está organizado como segue. A seção II apresenta a fundamentação teórica abordando a linguagem NorMAS-ML e o *framework* i* normativo. O mapeamento entre i* normativo e NorMAS-ML é detalhado na seção III. Na seção IV, dois exemplos de modelagem são apresentados. As discussões acerca do mapeamento e a sua comparação com os trabalhos relacionados são descritos na seção V. Finalmente, as considerações finais são abordados na seção VI.

## II. Fundamentação Teórica

### A. NorMAS-ML

A *Normative Multiagent System Modeling Language* (NorMAS-ML) [3] é uma linguagem de modelagem que permite o *design* de SMAN. Ela surgiu por meio da extensão da MAS-ML para possibilitar a modelagem das entidades típicas encontradas nos SMAN (ambiente, organização, agente, papel de agente, objeto e papel de objeto) em conjunto com os seguintes elementos que compõem uma norma [2]:

Este trabalho faz parte do Projeto número 5925 intitulado "Integração de Requisitos Organizacionais à Modelagem de Sistemas Multiagente Normativos" do Edital IFCE/PIBIC Jr/2017.

Thiago Gomes de Souza foi bolsista de Iniciação Científica Júnior do Instituto Federal do Ceará (IFCE/*Campus* Iguatu) (e-mail: thiagogsouza91@gmail.com).

Henrique Rabelo da Silva é bolsista de iniciação Científica Júnior do Instituto Federal do Ceará (IFCE/*Campus* Morada Nova) (e-mail: henriquersyt@gmail.com).

Emmanuel Sávio Silva Freire é mestre em Ciências da Computação e professor do Instituto Federal do Ceará (IFCE/*Campus* Morada Nova) (e-mail: savio.freire@ifce.edu.br).

- **Conceitos Deônticos**, descrevem as restrições de comportamento para os agentes na forma de obrigações, permissões e proibições;
- **Entidades Envolvidas**, que são as entidades que possuem o seu comportamento restringido por uma norma;
- **Ações**, que são as ações que estão sendo reguladas pela norma;
- **Restrições de Ativação**, que indicam a restrição ou um conjunto de restrições que ativam uma norma;
- **Sanções**, que são punições/recompensas que uma entidade recebe quando uma norma é violada/seguida, respectivamente;
- **Contexto**, indica a área de atuação (um ambiente ou uma organização) de uma norma.

Adicionalmente, Freire et al. [3] definiram um novo diagrama estático que permite a modelagem de normas. Dois exemplos utilizando o diagrama de normas podem ser encontrados na seção IV. As entidades que podem participar do diagrama são: (i) classe de norma, (ii) classe, (iii) classe de papel de objeto, (iv) classe de agente, (v) classe de papel de agente, (vi) classe de organização e (vii) classe de ambiente. Além disso, dos relacionamentos definidos na UML, também podem ser utilizados os seguintes relacionamentos no diagrama: (i) *ownership*, (ii) *play*, (iii) *inhabit*, (iv) *context*, (v) *restrict*, e (vi) *sanction*.

### B. *Framework i* Normativo*

O *framework iStar* normativo ou i* normativo [7] é uma extensão do *framework* i* [6] que possibilita a representação de leis e normas utilizando o modelo de ator, objetivo e dependências existente em i*. Para tanto, os autores consideraram as seguintes propriedades das normas relevantes para a aquisição de requisitos:

- **A relação de compromisso normativo (*the normative commitment relation*),** essa relação é formada por quem criou a norma (*source*), por aquele que é endereçado pela norma (*addressee*) e pelo artefato que será restringido pela norma (*legal artefact*);
- **O esquema de uma norma (*the schema of the norm*),** que está relacionado com o padrão de comportamento que a norma impõe para o *addressee*, como por exemplo, formas de atuação, objetivos e princípios a serem adotados;
- **As intenções de conformidade (*the compliance intentions*),** que estão associadas ao impacto real da norma sobre os *stakeholders*, identificando o que eles colocaram em ação para cumprir a norma.

Adicionalmente, os modelos que permitem que os engenheiros de requisitos possam modelar os *stakeholders* como atores e suas intenções como objetivos também foram alterados por conta da extensão. Esses modelos são:

- **O modelo de dependência estratégica (*Strategic Dependency* – SD)**, que foca na relação entre os agentes organizacionais;
- **O modelo estratégico de razão (*Strategic Rational* – SR)**, que possibilita a modelagem detalhada das relações intencionais internas de cada agente organizacional.

Dois exemplos de modelagem utilizando o *framework* i* normativo podem ser encontrados na seção IV.

## III. Mapeamento Entre i* Normativo e o Diagrama NorMAS-ML de Normas

Nesta seção, são apresentadas as diretrizes de mapeamento entre i* normativo e o diagrama NorMAS-ML de normas juntamente com exemplos para cada diretriz.

### A. *Diretrizes de Mapeamento*

As diretrizes de mapeamento determinam as regras pelas quais a associação entre o modelo i* normativo e o diagrama NorMAS-ML de normas é realizada corretamente. Para tanto, foram modificadas as diretrizes propostas por Alencar et al. [9] e Melo et al. [10] que possibilitavam o mapeamento entre i* e o diagrama UML de classes.

O objetivo das diretrizes é manter a consistência e o rastreamento entre o sistema a ser desenvolvido e os objetivos da organização, considerando as normas que regulam o comportamento das entidades que compõem os SMAN. Desta forma, aumenta-se as chances do sistema que está sendo desenvolvido atender os objetivos da organização de uma forma mais precisa. Além disso, pode-se relacionar os artefatos gerados na fase de análise (modelo i* normativo) com os gerados na fase de projeto (diagrama NorMAS-ML de normas).

As diretrizes foram subdivididas nos seguintes sete grupos: (i) **D1**, mapeamento de atores; (ii) **D2**, mapeamento de tarefas; (iii) **D3**, mapeamento dos recursos; (iv) **D4**, mapeamento de objetivos e objetivos *soft*; (v) **D5**, mapeamento de relação entre meios-fim (*means-end*); (vi) **D6**, mapeamento de decomposição de tarefas; e (vii) **D7**, mapeamento de normas. A TABELA I apresenta as diretrizes de mapeamento entre i* Normativo e o diagrama NorMAS-ML de normas.

TABELA I. Diretrizes de mapeamento entre i* normativo e o diagrama NorMAS-ML de normas

| Tipo | Número | i* Normativo | Diagrama NorMAS-ML de Normas |
|---|---|---|---|
| D1 | D1.1 | Papel | Classe de Papel de Agente |
| | D1.2 | Agente | Classe de Agente |
| | D1.3 | Posição | Classes de Organização |
| | D1.4 | Relacionamento IS-PART-OF entre papel e posição | Relacionamento *ownership* entre Classes de Papel de Agente e Organização |
| | D1.5 | Relacionamento IS-PART-OF entre posições | Relacionamento *aggregation* entre Classes de Posição |
| | D1.6 | Relacionamento IS-PART-OF entre agente e posição | Não apresenta correspondente |
| | D1.7 | Relacionamento IS-A | Generalização/Especialização de classes de Papel de Agente ou entre Classes de Agentes ou de Organizações |

| | | | |
|---|---|---|---|
| | D1.8 | Relacionamento i* *OCCUPIES* entre um agente e uma posição | Não apresenta correspondente |
| | D1.9 | Relacionamento i* *COVERS* entre uma posição e um papel | Relacionamento *ownership* entre Classes de Papel de Agente e Organização |
| | D1.10 | Relacionamento i* *PLAYS* entre um agente e um papel | Relacionamento *play* entre Classes de Agente e de Papel de Agente |
| **D2** | D2.1 | Tarefa no modelo SD | Atributo *agent action* com visibilidade pública ou *resource*, caso seja uma norma. |
| | D2.2 | Tarefas no modelo SR | Atributo *agent action* com visibilidade privada ou *resource*, caso seja uma norma. |
| **D3** | D3.1 | Recursos definidos no modelo SD | Classe de Objeto, se essa dependência tem a característica de um objeto |
| | D3.1 | Recursos definidos no modelo SD | Atributo *agent action* com visibilidade privada |
| | D3.2 | Recursos (sub-recursos) definidos no modelo SR | Atributo *agent action* com visibilidade privada na classe que representa o ator em que o sub-recurso pertence (se esse sub-recurso não pode ser entendido como um objeto). |
| | D3.2 | Recursos (sub-recursos) definidos no modelo SR | Classe de Objeto, se esse sub-recurso tem a característica de um objeto |
| **D4** | D4.1 | Objetivo (ou objetivo *soft*) definido no modelo SD | Atributo *goal* com visibilidade pública na classe da entidade correspondente. |
| | D4.2 | Objetivo (ou objetivo *soft*) definido no modelo SR | Atributo *goal* com visibilidade pública na classe da entidade correspondente |
| **D5** | D5.1 | Objetivos (ou objetivos *soft*) - Objetivos (ou objetivos *soft*) | Representada por regras (podem ser descritas em OCL) da entidade correspondente |
| | D5.2 | Objetivos (ou objetivos *soft*)- Tarefa, Recurso-Tarefa | Representada por regras (podem ser descritas em OCL) da entidade correspondente |
| | D5.3 | Tarefa-Tarefa | Representada por regras (podem ser descritas em OCL) da entidade correspondente |
| **D6** | D6.1 | Decomposição de tarefa no modelo SR | Representada pelas pré e pós-condições (podem ser expressas em OCL) da *agent action* correspondente |
| **D7** | D7.1 | Fonte da Norma | Relacionamento *context* entre Classes de Norma e de Papel de Agente |
| | D7.2 | Norma | Classe de Norma de Obrigação |
| | D7.3 | Endereçado pela Norma | Relacionamento *restrict* entre Classes de Norma e de Agente ou entre Classes de Norma e de Papel de Agente ou entre Classes de Norma e de Organização. |
| | D7.4 | Prescrições da Norma | Segue as diretrizes D2, D3, D4, D5 e D6. |

### *B. Detalhamento das Diretrizes*

Nesta seção, as diretrizes de mapeamento serão detalhadas por meio da discussão conceitual dos elementos mapeados e da apresentação da transformação das entidades de i* normativo para seu correspondente no diagrama NorMAS-ML de normas. De acordo com a TABELA I , a **diretriz D1.1** menciona que um papel deve ser transformado em uma classe de Papel de Agente. Em i* normativo, um papel é uma caracterização abstrata do comportamento de um ator dentro de algum contexto. Esse conceito é equivalente em NorMAS-ML, pois um papel de agente é responsável por restringir e orientar o comportamento de agentes quando executam o papel. A Fig. 1 (a) apresenta essa transformação.

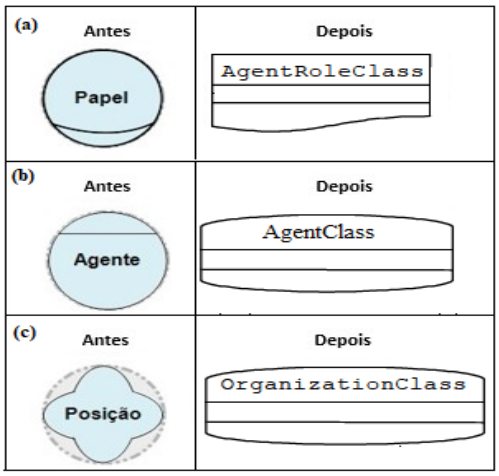


Fig. 1. Transformação de atores de i* normativo para entidades de NorMAS-ML

Considerando o conceito de agente em i* normativo é equivalente em NorMAS-ML, a **diretriz D1.2** indica que um agente em i* normativo deve ser transformado em uma classe de agente no diagrama de normas (veja a Fig. 1 (b)). Em relação à entidade posição, no *framework* i* normativo, representa um conjunto de papéis normalmente executado por agentes. Esse conceito é encontrado na entidade Organização de NorMAS-ML. Uma organização é capaz de definir papéis que devem ser exercidos pelos agentes. Logo, possui um conjunto de papéis. Consequentemente, a **diretriz D1.3** determina que uma posição em i* normativo seja transformado em uma classe de organização no diagrama de normas. A Fig. 1 (c) ilustra essa transformação.

A **diretriz D1.4** determina que o relacionamento *IS-PART-OF* entre uma posição e um papel seja mapeado para o relacionamento *ownership* de NorMAS-ML. Ele indica que um proprietário (organização) conheça todos os seus membros (papéis de agente) (Veja a Fig. 2). Além disso, a **diretriz D1.5** indica que o relacionamento *IS-PART-OF* entre posições seja transformado no relacionamento *aggregation* de NorMAS-ML. Vale ressaltar que, em NorMAS-ML, não há correspondente para o mapeamento do relacionamento *IS-PART-OF* entre uma posição e um agente (**diretriz D1.6**), pois esse relacionamento é feito de forma implícita quando um agente executa um papel de agente pertencente a uma organização.

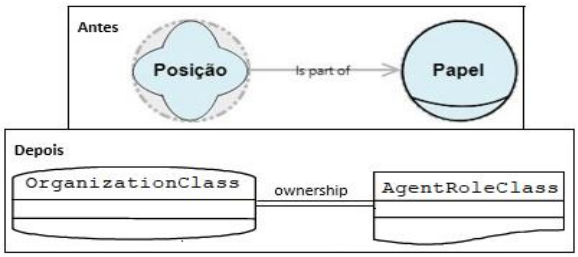


Fig. 2. Transformação do relacionamento IS-PART-OF em ownership de NorMAS-ML

A **diretriz D1.7** indica que o relacionamento *IS-A* deve ser transformado em um relacionamento *specialization* em NorMAS-ML, pois indica relação de herança entre as entidades participantes. Esses relacionamentos possuem a mesma semântica. A Fig. 3 apresenta essa transformação. Em relação à **diretriz D1.8**, não é possível representar o relacionamento *OCCUPIES* em NorMAS-ML pelo mesmo motivo apresentado para a diretriz D1.5.

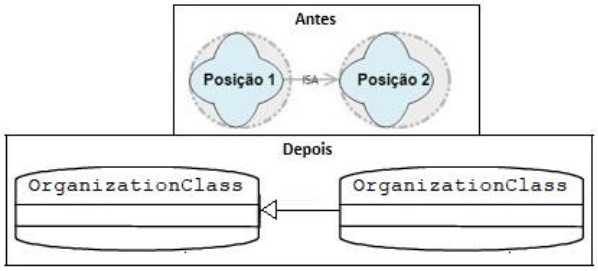


Fig. 3. Transformação do relacionamento IS-A em specialization de NorMAS-ML

Considerando que o relacionamento *COVERS* de i* normativo indica a relação entre uma posição e seus papéis, a **diretriz D1.9** estabelece que esse relacionamento seja mapeado para o relacionamento *ownership* entre classes de papel de agente e de organização. A Fig. 4 apresenta esse mapeamento. Em relação ao relacionamento *PLAYS* de i* normativo, a **diretriz D1.10** determina que ele seja transformado no relacionamento play de NorMAS-ML, pois apresentam a mesma semântica. A Fig. 5 ilustra essa transformação.

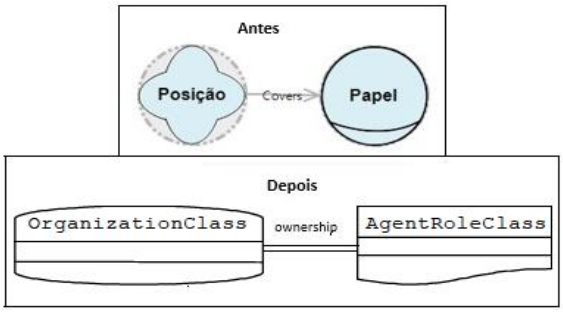


Fig. 4. Transformação do relacionamento COVERS em ownership de NorMAS-ML

É importante ressaltar que os relacionamentos *IS-PART-OF* e *COVERS* de i* normativo foram mapeados para o relacionamento *ownership* de NorMAS-ML, conforme as diretrizes D1.4 e D1.9, respectivamente. Assim, por mais que *IS-PART-OF* e *COVERS* tenham semânticas diferentes no modelo i*, o relacionamento *ownership* garante tanto a associação entre atores (organização e papéis de agente) demostrando que um ator faz parte de outro ator e a associação entre uma posição (organização) e seus papéis de agente.

A **diretriz D2.1** indica que as tarefas definidas no modelo SD devem ser convertidas em *agent actions* com visibilidade pública na classe do ator que relaciona com essas tarefas. Além disso, a **diretriz D2.2** estabelece que as tarefas definidas no modelo SR também devem se tornar *agent actions*, porém com visibilidade privada (Veja a Fig. 6). É importante ressaltar que um *agent action* de NorMAS-ML corresponde a uma ação que pode ser executada por um agente, por uma organização ou por um papel de agente.

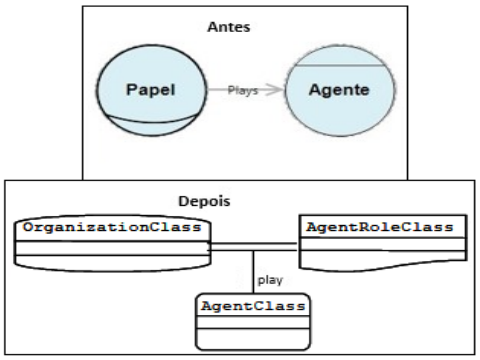


Fig. 5. Transformação do relacionamento PLAYS em play de NorMAS-ML

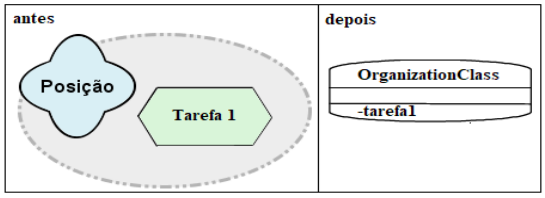


Fig. 6. Transformação de tarefas em agent actions de NorMAS-ML

A **diretriz D3.1** afirma que os recursos feitos no modelo SD devem ser transformados em um objeto em NorMAS-ML, caso essa dependência tenha característica de objeto. Entretanto, se a dependência não tiver essa característica, o recurso deve ser transformado em um *agent action* com visibilidade privada na classe do ator. O mesmo ocorre para os recursos do modelo SR conforme a **diretriz D3.2**. A Fig. 7 apresenta esses mapeamentos.

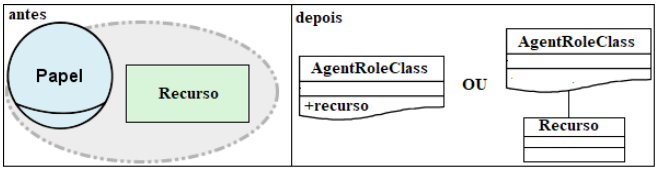


Fig. 7. Transformação de recursos em objetos ou agent actions de NorMAS-ML

Em relação aos objetivos no modelo SD e SR, as **diretrizes D4.1 e D4.2** indicam que devem ser transformados em *goals* em NorMAS-ML com visibilidade pública na classe do ator correspondente. Um atributo *goal* representa o objetivo de um agente, de uma organização ou de um papel de agente em NorMAS-ML. A Fig. 8 ilustra essa transformação.

A **diretriz D5** refere-se aos meios-fim (*means-end*), que sugere que pode-se ter formas alternativas para alcançar um objetivo. Assim, as **diretrizes D5.1, D5.2 e D5.3** indicam que deve ser criada uma nota contendo a instrução correspondente, vinculada à classe do ator. A **diretriz D6** está relacionada com

a decomposição de tarefas. Essa decomposição indica o que deve ser feito para a realização de uma determinada tarefa. Assim, por meio da **diretriz D6.1**, a decomposição deve também ter uma nota vinculada à classe do ator. A Fig. 9 apresenta essas transformações de *means-end* e de decomposição de tarefas.

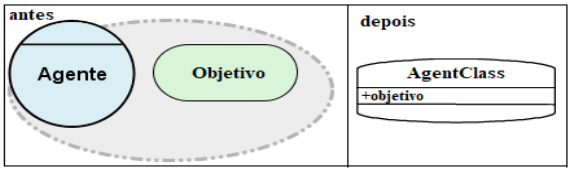


Fig. 8. Transformação de recursos em objetos ou agent actions de NorMAS-ML

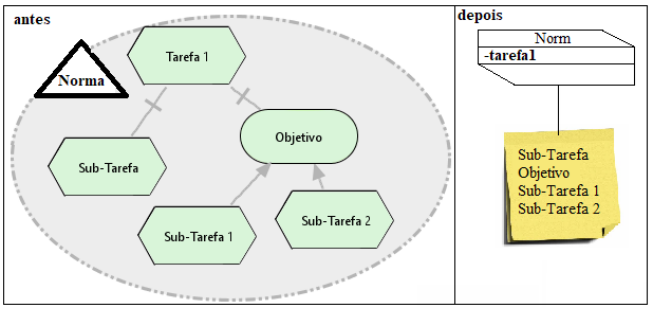


Fig. 9. Transformação de means-end e de decomposição de tarefas em NorMAS-ML

A **diretriz D7.1** indica que a fonte da norma deve ser transformada para o relacionamento *context* de NorMAS-ML entre classes de norma e de papel de agente. A Fig. 10 apresenta esse mapeamento. Além disso, a **diretriz D7.2** estabelece que a norma no *framework* i* normativo deve ser transformada na entidade norma de NorMAS-ML com conceito deôntico de obrigação (Veja a Fig. 10). Foi utilizado esse conceito deôntico pois em i* normativo não é possível a diferenciação entre normas de obrigação, permissão e proibição no seu metamodelo.

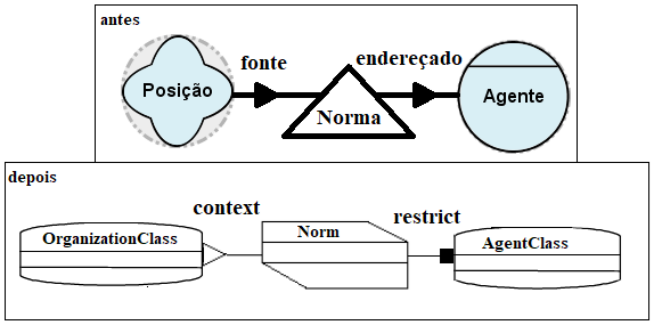


Fig. 10. Transformação da norma e seus elementos em NorMAS-ML

A **diretriz D7.3** afirma que a entidade endereça pela norma deve ser representa pelo relacionamento *restrict* de NorMAS-ML (Veja a Fig. 10). Assim, pode-se vincular classes de agente, de papel de agente ou de organização a uma classe de norma. Por outro lado, a **diretriz D7.4** estabelece que as prescrições da norma devem seguir as diretrizes D2, D3, D4, D5 e D6 (Veja a Fig. 9).

## IV. Exemplos de Modelagem

Para ilustrar o uso das diretrizes propostas na seção anterior, foram escolhidos dois exemplos de modelagem: (i) o sistema *Meeting Scheduler* definido e modelado por Yu [11] utilizando i* e o sistema *SmartCD* definido por Alencar et al. [9]. Além disso, foram definidas e incluídas normas para esses sistemas para possibilitar a utilização das diretrizes de mapeamento entre i* normativo e o diagrama NorMAS-ML de normas.

### A. Sistema Meeting Scheduler

O sistema *Meeting Scheduler* permite a realização de agendamento de reuniões. Inicialmente, Yu [11] definiu três atores: *Meeting Initiator*, *Meeting Participant* e *Important Participant*. A Fig. 11 apresenta o modelo *Strategic Dependence* (SD) definido por esse autor.

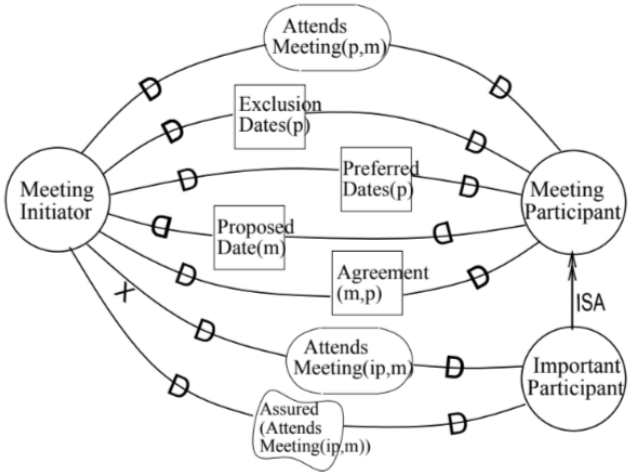


Fig. 11. Modelo *Strategic Dependence* (SD) para o *Meeting Scheduler* [11]

Adicionalmente, a posição *Meeting Scheduler* definidas as seguintes normas: (i) N1: o agente *Meeting Initiator* deve organizar reunião antes da data da reunião; e (ii) N2: o agente *Meeting Participant* deve estar presente no dia da reunião (*participationInMeeting*). A Fig. 12 apresenta o modelo *Strategic Rational* (SR) considerando os atores *Meeting Initiator*, *Meeting Scheduler* e *Meeting Participant* em conjunto com as normas apresentadas anteriormente. É importante salientar que, no trabalho de Yu [11] as entidades *Meeting Initiator*, *Meeting Scheduler* e *Meeting Participant* foram definidas como atores, porém foram adaptadas para posição, agente e agente, respectivamente. Essa decisão de projeto foi necessária porque os atores em NorMAS-ML podem ser posições, agentes ou papeis, conforme a Fig. 1.

Após a transformação manual de modelos, foi gerado o diagrama NorMAS-ML de normas apresentado na Fig. 13, considerado as diretrizes de mapeamento definidas na TABELA I. No diagrama, os agentes *Meeting Initiator* e *Meeting Participant* e a posição *Meeting Scheduler* foram transformados em classes de agente e de organização, respectivamente, seguindo as **diretrizes D1.2** e **D1.3**. As

tarefas foram mapeadas para *agentActions* nas classes das entidades responsáveis por cada tarefa (**diretriz D2**). Conforme as **diretrizes D5 e D6**, os links de decomposição de tarefas e de *means-end*, foram mapeados para um comentário associado à organização *Meeting Scheduler* e aos agentes *Meeting Initiator* e *Meeting Participant*. Em relação as normas, N1 e N2 foram mapeadas para classes de norma (**diretriz D7.2**), suas fontes foram representadas pelo relacionamento *context* (**diretriz D7.1**) e seus endereçamentos pelo relacionamento *restrict* (**diretriz D7.3**). Finalmente, as prescrições de cada norma foram mapeados para *resource* seguindo a diretriz **D7.4**.

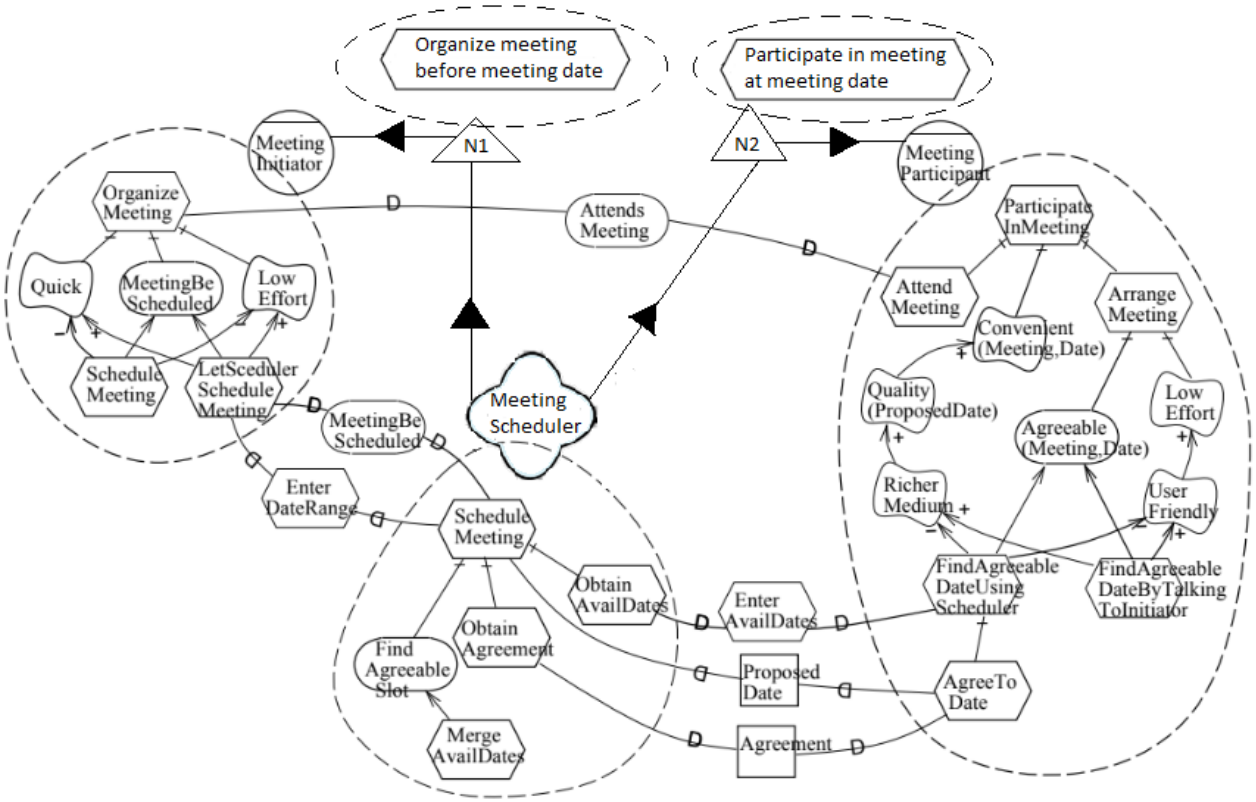


Fig. 12. Modelo *Strategic Rational* (SR) para o *Meeting Scheduler* (Adaptado de [11])

### B. *Sistema SmartCD*

O sistema *SmartCD* permite a realização de pedidos por meio de um catálogo online. Inicialmente, Melo et al. [10] definiram três posições: *Client*, *Store Management* e *SmartCD*. A última posição corresponde ao sistema a ser desenvolvido que deve tratar as encomendas, notificar as chegadas de CD e fornecer o catálogo. A Fig. 14 apresenta o modelo *Strategic Dependence* (SD) definido por esses autores. Além disso, para a posição *SmartCD*, foram definidas as seguintes normas: (i) N1: A posição *SmartCD* definiu que a posição *Internet Sales* deve registrar o cliente antes de efetuar a venda de CDs; (ii) N2: A posição *SmartCD* definiu que a posição *Inventory* deve atualizar o estoque sempre que houver uma nova venda; e (iii) N3: A posição *Internet Sales* definiu que o ator *Office Boy* possui um prazo de duas horas, obrigatoriamente, para a entrega de CDs na cidade que este ator se encontra.

A Fig. 15 apresenta o modelo *Strategic Rational* (SR) considerando a posição *SmartCD* em conjunto com as normas apresentadas acima. Esse diagrama foi baseado no exemplo de modelagem apresentado em Melo et al. [10]. Após a transformação manual de modelos, foi gerado o diagrama NorMAS-ML de normas apresentado na Fig. 16. No diagrama, os papéis *CDReservation* e *CDDelivery*, o agente *OfficeBoy*, e as posições *Inventory*, *SmartCD*, *Financial*, *InternetSales* e *Client* foram transformados em classes de papel de agente, de agente e de organização, respectivamente, seguindo as **diretrizes D1.1, D1.2 e D1.3**. Além disso, os relacionamentos *IS-PART-OF* foram transformados em agregações entre a organização *SmartCD* e as organizações *Inventory* e *Financial*, conforme a **diretriz D1.5**. O relacionamento *COVERS* foi transformado no relacionamento *ownership* (**diretriz D1.9**) entre a organização *InternetSales* e os papéis de agente *CDReservation* e *CDDelivery*. O relacionamento *PLAYS* foi transformado no relacionamento *play* de NorMAS-ML (**diretriz D1.10**) entre o papel de agente *CDDelivery* e o agente *OfficeBoy*. Vale ressaltar que o relacionamento *OCCUPIES* não foi mapeado para NorMAS-ML (**diretriz D1.8**), mas as entidades ligadas por esse relacionamento foram mapeadas para o diagrama de normas.

As tarefas foram mapeadas para *agentActions* nas classes das entidades responsáveis por cada tarefa (**diretriz D2**). O recurso *PersonalData* foi mapeado como uma classe no diagrama, pois apresentava característica de objeto (**diretriz D3.2**). Conforme as **diretrizes D5 e D6**, os *links* de

decomposição de tarefas e de *means-end*, foram mapeados para um comentário associado à organização *InternetSales*. As normas, N1, N2 e N3 foram mapeadas para classes de norma (**diretriz D7.2**), suas fontes foram representadas pelo relacionamento *context* (**diretriz D7.1**) e seus endereçamentos pelo relacionamento *restrict* (**diretriz D7.3**). Finalmente, as prescrições de cada norma foram mapeados para *resource* seguindo a **diretriz D7.4**.

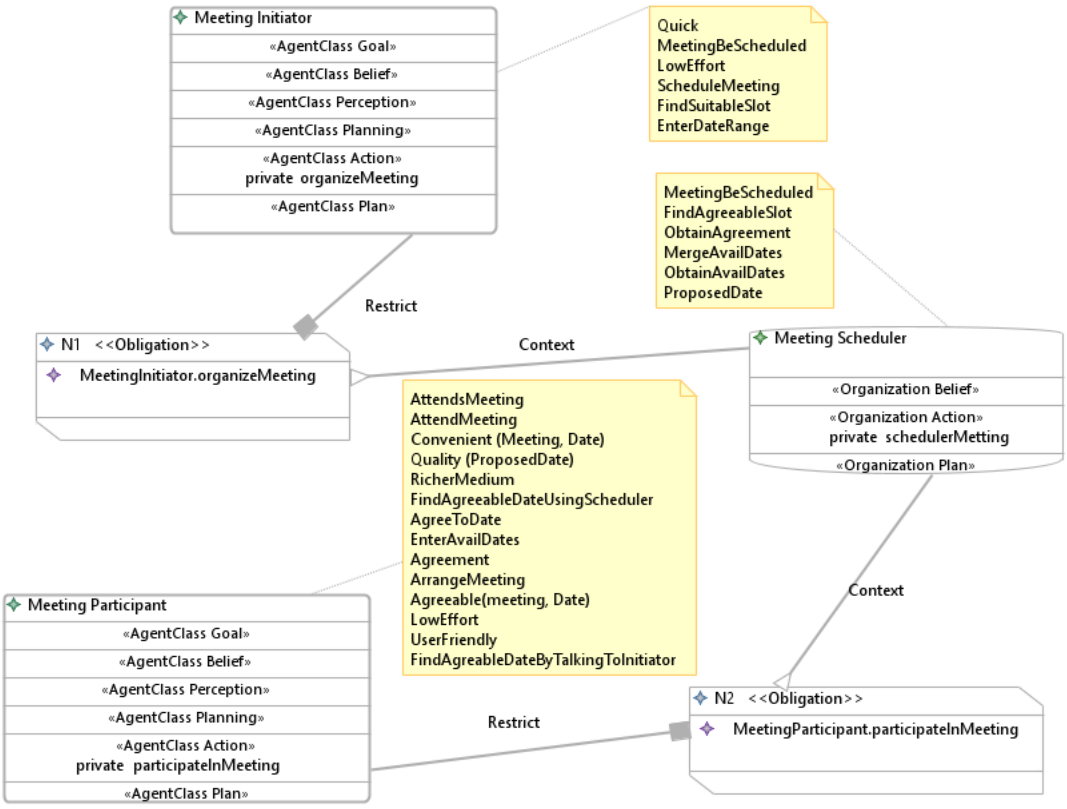


Fig. 13. Diagrama NorMAS-ML de Normas para o *Meeting Scheduler* (Fonte: Próprios Autores)

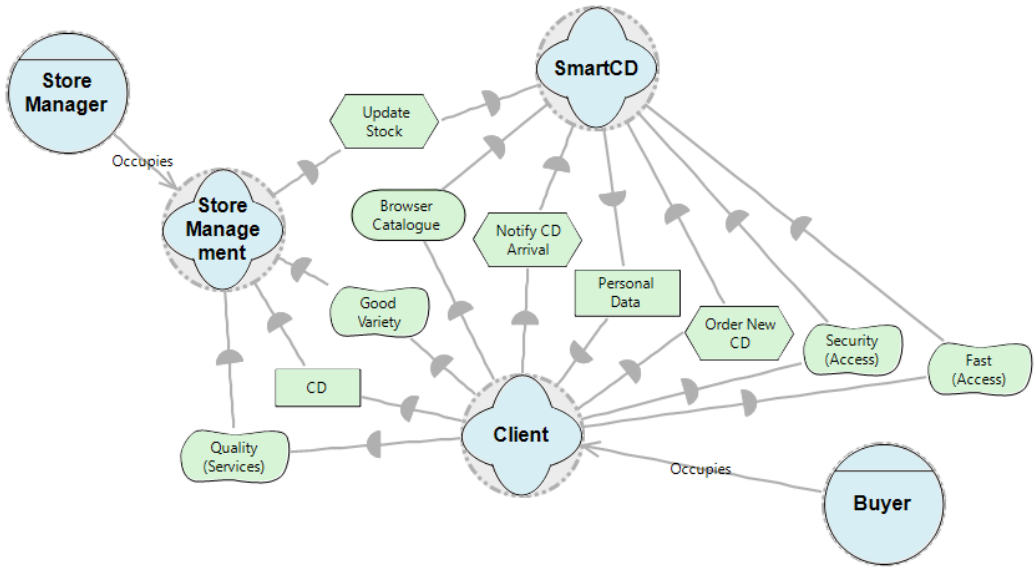


Fig. 14. Modelo *Strategic Dependence* (SD) para o *SmartCD* [10]

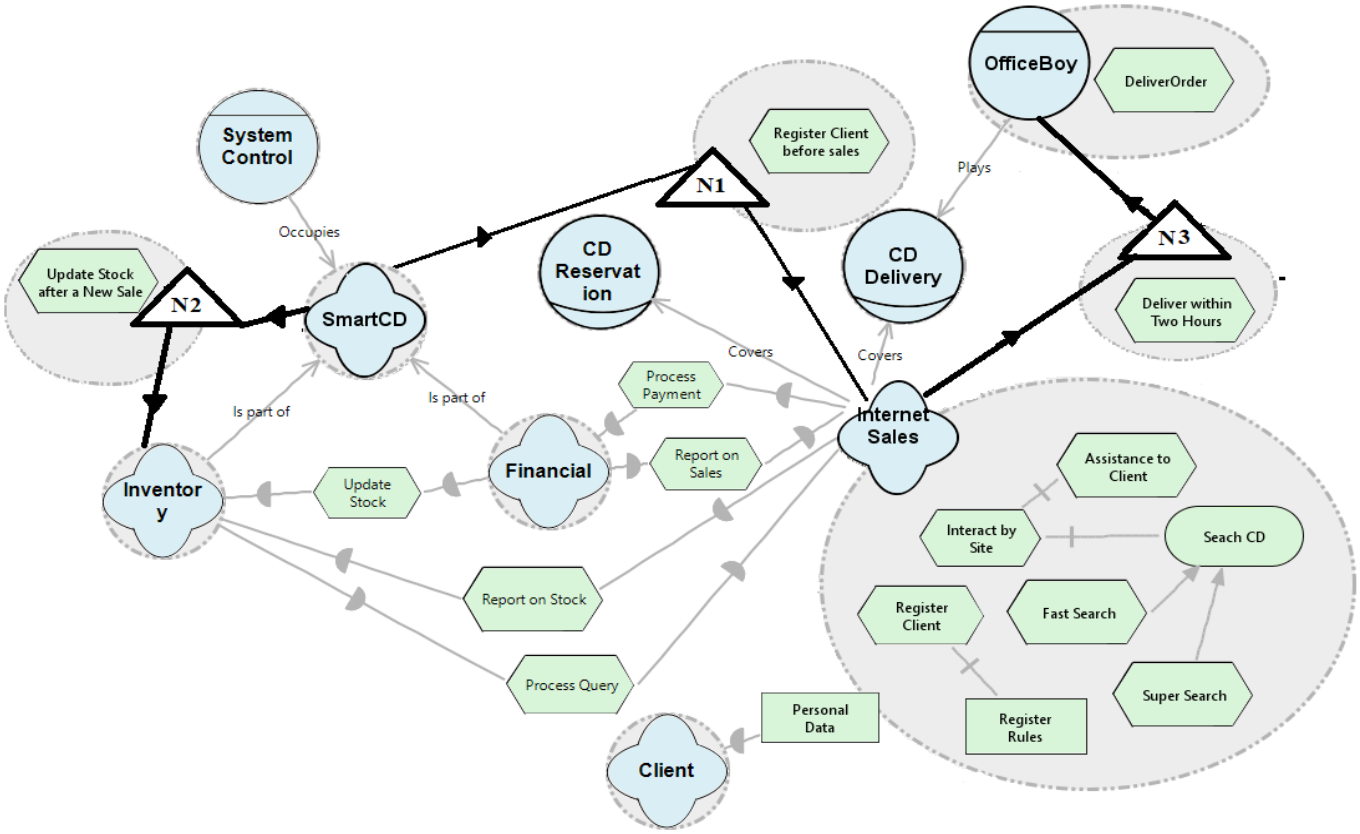


Fig. 15. Modelo *Strategic Rational* (SR) para o *SmartCD* (Fonte: Próprios Autores)

## V. DISCUSSÃO

Por meio das diretrizes de mapeamento apresentadas na seção III, é possível transformar modelos i* normativo para o diagrama NorMAS-ML de normas. Essa transformação de modelos foi exemplificada na seção IV por meio da modelagem dos sistemas *Meeting Scheduler* e *SmartCD*. Como vantagens, pode-se citar:

- O analista de requisitos pode concentra-se no entendimento das necessidades dos usuários do sistema a ser desenvolvido e mapeá-las para os modelos SD e SR de i* normativo. Além disso, as normas que regulam o comportamento das entidades desse sistema também podem ser modelados;
- O projetista pode utilizar as diretrizes de mapeamento propostas no presente trabalho para auxiliar na transformação dos modelos i* normativo no diagrama NorMAS-ML de normas. Com isso, o projetista se concentrará na identificação de possíveis melhorias para o diagrama em conjunto com os analistas de requisitos;
- A integração entre as fases de requisitos e de modelagem permite que os erros associados à transformação de modelos entre essas fases seja diminuído, possibilitando uma maior consistência entre os modelos gerados.

Entretanto, essa pesquisa apresenta as seguintes limitações:

- Não foi proposta uma ferramenta para automatizar a transformação de modelos i* normativo para o diagrama de normas. Por consequência, deve ser feita de forma manual;
- Ao transformar os modelos i* normativo para o diagrama NorMAS-ML de normas, o projetista precisaria seguir as diretrizes e realizar a transformação de forma manual. Logo, possíveis inconsistências poderiam ser encontradas nesse processo.

Em relação aos trabalhos relacionados, vários autores [12][13] propuseram métodos que permitem a transformação entre modelos. Mais especificamente, as pesquisas de Lucena et al. [14] e Aguilar et al. [15] propuseram a transformação de modelos i* para modelos arquitetônicos Acme e para a linguagem QVT, respectivamente.

Em relação ao escopo do presente trabalho, as abordagens propostas por [9] [10] [16] e [17] se aproximam ao objetivo deste trabalho pelo fato de propor o mapeamento de i* para diagramas da UML. No entanto, nenhum dos trabalhos descritos acima consideram o contexto dos SMAN, pois não abordam as normas que regulam as entidades desses sistemas. Com isso, o presente trabalho se diferencia dos demais por utilizar o *framework* i* normativo em conjunto com o diagrama NorMAS-ML de normas, possibilitando a integração das fases de requisitos e modelagem via modelo.

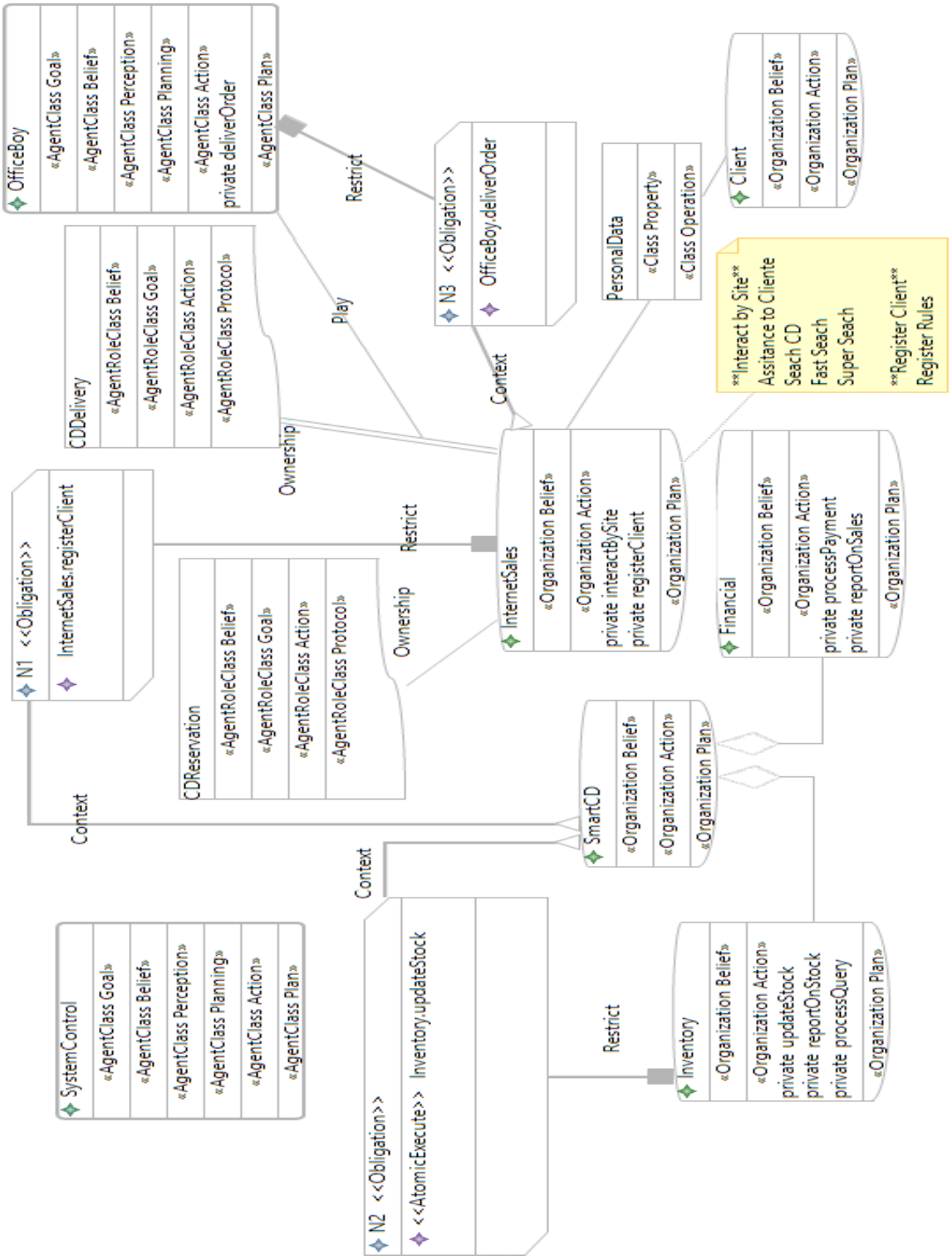


Fig. 16. Diagrama NorMAS-ML de Normas para o *SmartCD* (Fonte: Próprios Autores)

## VI. Considerações Finais

Este artigo teve como objetivo propor um conjunto de diretrizes para possibilitar o mapeamento entre as entidades existentes em i* normativo e NorMAS-ML, identificando quais as semelhanças e as diferenças conceituais existentes entre eles. Para tanto, foram analisadas conceitualmente as entidades previstas em i* normativo e do diagrama NorMAS-ML de

normas. Assim, foi proposto um conjunto de diretrizes para a transformação de modelos entre esse *framework* e essa linguagem. Adicionalmente, dois exemplos de modelagem, baseado nos sistemas *Meeting Scheduler* e *SmartCD*, foram realizados com o intuito de ilustrar as diretrizes de transformação propostas.

Além das diretrizes de transformação, a contribuição desse artigo foi a integração entre as fases de requisitos e de modelagem previstas no processo de desenvolvimento de software, considerando o paradigma orientado a agentes. Mais especificamente, essa integração colabora para que o entendimento de sistemas complexos (SMAN) seja facilitada por meio da utilização de linguagem visual (diagramas) e para a diminuição de inconsistências que normalmente ocorrem na transição de fases do processo de desenvolvimento de *software*.

Como trabalhos futuros, utilizando as diretrizes propostas neste trabalho, pode-se destacar:

- A formalização das diretrizes baseando na abordagem de Melo et al. [10];
- A implementação de uma ferramenta capaz de ler os modelos em i* normativo e gerar o diagrama NorMAS-ML de normas correspondente.

## REFERÊNCIAS


[1] G. Boella, L. van der Torre and H. Verhagen, “Introduction to normative multiagent systems”, Computational & Mathematical Organization Theory, vol. 12, no. 2, pp. 71–79.

[2] V. Silva, C. Braga and K. Figueiredo, “A Modeling Language to Model Norms”. In: Workshop on Coordination, Organization, Institutions and Norms in agent systems at International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS10), Toronto, p. 25-32.

[3] E. S. S. Freire, M. I. Cortés, M. I., E.J. T. Gonçalves and Y. S. Lopes, “NorMAS-ML: A Modeling Language to Model Normative Multi-Agent Systems”. In: 14th International Conference on Enterprise Information Systems (ICEIS), 2012, Wroclaw (Poland).

[4] S. Franceto, “Especificação e Implementação de uma Ferramenta para Elicitação de Requisitos de Software baseada na Teoria da Atividade”. 2005. Disponível em: < http://goo.gl/4wj9EZ>. Acessado em 10 de outubro de 2017.

[5] I. Sommerville, Engenharia de Software. 9 ed. São Paulo: Pearson Addison- Wesley, 2011.

[6] E. Yu, “Social Modeling and i*”. In: Faculty of Information, University of Toronto, Toronto, Canada M5S 3G6, 2002.

[7] A. Siena, N. Maiden, J. Lockerbie, K. Karlsen, A. Perini and A. Susi, “Exploring the Effectiveness of Normative i* Modelling: Results from a Case Study on Food Chain Traceability”. In: CAiSE '08 Proceedings of the 20th international conference on Advanced Information Systems Engineering Pages 182 – 196.

[8] E. S. S. Freire and M. I. Cortés, “Integrando Requisitos Organizacionais à Modelagem de Sistemas Multiagente Normativos”. In: 10th Workshop-School on Agents, Environments, and Applications (WESAAC), 2016, Alagoas.

[9] F.M.R. Alencar, F. Pedroza, J. F. B. Castro and R. C. O. Amorim, “New Mechanisms for the Integration of Organizational Requirements and Object Oriented Modeling”, VI WORKSHOP DE ENGENHARIA DE REQUISITOS, Piracicaba-SP , 2003.

[10] J. Melo, A. Sousa, C. Agra, J. Júnior, J. Castro. and F. Alencar, “Formalization of Mapping Rules from iStar to Class Diagram in UML”. In: 29th Brazilian Symposium on Software Engineering (SBES), 2015, Belo Horizonte.

[11] E. Yu, “Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering”. In: Proceedings of ISRE '97: 3rd IEEE International Symposium on Requirements Engineering, pages 226-235.

[12] B. Richard, “A deidealisation semantics for KAOS”. In Proceedings of the 2010 ACM Symposium on Applied Computing, pages 267-274. ACM, March 2010.

[13] Z. Sun, J. Wang, K. He, S, Xiang and D. Yu, “A Model Transformation Method in Service-Oriented Domain Modeling”. In Proceedings of the 2010 21st Australian Software Engineering Conference, pages 107-116. IEEE Computer Society, 2010.

[14] M. Lucena, J. Castro, C. Silva, F. Alencar and E. Santos, “Stream: a strategy for transition between requirements models and architectural models”. In Proceedings of the 2011 ACM Symposium on Applied Computing, pages 699-704. ACM, 2011.

[15] J. Aguilar, I. Garrigós, J. Mazón and J. Trujillo, “An MDA approach for goal-oriented requirement analysis in Web engineering”, J. Univers. Comput. Sci., vol. 16, no. 17, pp. 2475–2494, 2010.

[16] F. Alencar, G. Gianchetti and O. Pastor, “From i* requirements models to conceptual models of a Model Driven Development process”. In: The Practice of Enterprise Modeling - Second IFIP WG 8.1 Working Conference, PoEM 2009, Proceedings, pp. 99-114.

[17] C. G. Filho, A. Zisman and G. Spanoudakis, “Traceability Approach for i* and UML Models”. In: Proceedings of 2nd International Workshop on Software Engineering for Large-Scale Multi-Agent Systems (SELMAS’03), Portland.