Uma Notação Textual Modular e Escalável para Modelos de Requisitos iStar

Fábio Penha[1,2], Márcia Lucena[2], Leonardo Lucena[1], Fernanda Alencar[3] e Celso Agra[3]

[1] Instituto Federal do Rio Grande do Norte - IFRN, Natal, Brasil
[2] Universidade Federal do Rio Grande do Norte - UFRN, Natal, Brasil
[3] Universidade Federal de Pernambuco - UFPE, Recife, Brasil

fabio.penha,leonardo.lucena@ifrn.edu.br,marciaj@dimap.ufrn.br
fernanda.ralencar@ufpe.br, celso.agra@gmail.com

Resumo O framework iStar é uma notação gráfica utilizada para modelar dependências e raciocínios estratégicos em requisitos de sistemas computacionais. Um desafio recorrente desta notação está relacionado com a falta de modularidade e por consequência, escalabilidade. Uma das principais reflexo destas fraquezas é a dificuldade em lidar com modelos mais complexos. Neste contexto, este artigo apresenta uma notação textual, modular e escalável, para modelos de requisitos iStar capaz de lidar com modelos complexos. A notação proposta também propõe uma orientação para leitura e entendimento dos modelos iStar por todos os envolvidos, facilitando a análise e verificação dos modelos.

Palavras-chave: framework istar; modularidade; escalabilidade; notação textual.

1 Introdução

Com a expansão tecnológica [1], a sociedade e as organizações estão cada vez mais complexas, sociais e técnicos. Neste cenário, os sistemas computacionais tendem a continuar crescendo em escala e complexidade [2]. Isto torna o processo de modelagem uma atividade crucial para que os softwares consigam gerar valor ao ambiente em questão. O trabalho apresentado por Eric Yu [3] mostra uma ontologia capaz de capturar, do universo da informação, as dimensões sociais e intencionais que eram pouco exploradas no ambiente organizacional, quando comparado com as dimensões estáticas e dinâmicas já consolidadas pela comunidade de engenharia de software [4].

Apesar da relevância acadêmica e utilização em diversas áreas, tais como: telecomunicações, controle de tráfego aéreo, agricultura e saúde [4], os modelos iStar encontram resistência para uma maior aceitação na indústria. Esta resistência é relacionada com as limitações do framework iStar, como por exemplo, a falta de modularidade e escalabilidade [4,5,6,7,8,9,10]. Estas fraquezas tornam-se mais evidentes na modelagem de sistemas complexos; sistemas que

são compostos por um grande número de partes que interagem de maneira não simples, dadas as propriedades das partes e as leis de sua interação [11].

Apesar dos modelos gráficos mostrarem-se mais eficientes e intuitivos para análise e interpretação [4,12], a forma de como os modelos iStar foram concebidos impacta diretamente na escalabilidade, como consequência da falta de modularidade [13,8,9,10]. No estudo empírico apresentado em [14], foi analisada a compreensibilidade dos modelos gráficos e textuais. Nesta análise, assim como em [15], constataram que não existe nenhuma superioridade significativa entre ambos estilos. Para estes autores, a escolha por um determinado estilo deve ser feita através de critérios mais adequados, como: poder de expressividade, capacidade de análise formal, estética e usabilidade. A alegação seria a diversidade de público e suas habilidades na hora de analisar a modelagem realizada.

Este trabalho apresenta uma notação textual mais detalhada e evoluída da versão preliminar [16] exposta na comunidade iStar. A proposta esclarece o elemento modular dos modelos gráficos e uma notação textual mais sucinta, sintetizado aos elementos e seus relacionamentos. Além disso, este trabalho propõe uma orientação para leitura e entendimento dos modelos iStar por todos os envolvidos, com ou sem experiência nos modelos propostos pelo framework iStar.

O artigo está estruturado da seguinte forma: na seção seguinte é apresentado o framework iStar e as suas principais limitações; na seção 3, a notação textual e os requisitos que nortearam o seu desenvolvimento; na seção 4 é realizada uma aplicação de uso da notação textual; na seção 5, os trabalhos relacionados são expostos juntamente com discussões sobre a proposta; e finalmente a conclusão.

2 Framework IStar

O Framework IStar é uma abordagem da Engenharia Requisito Orientada a Objetivo. Para Franch [7], esta abordagem trouxe alguns atrativos que incentivaram muitos trabalhos nos últimos anos, por exemplo : uma abordagem (i) que, além de combinar os paradigmas orientados a objetivo e agente, (ii) possibilita uma atenção nas propriedades intencionais e nos relacionamentos ao invés dos comportamentos atuais. Estes atrativos podem prover uma rica expressividade que é bem apropriada para os conceitos sociais. O framework [3] articula uma notação de intencionalidades distribuídas em dois modelos: a modelo de SD (do inglês, Strategic Dependency) e o modelo SR (do inglês, Strategic Rational).

Modelo de Dependência Estratégica (SD): é uma rede de relacionamentos que mapeia as dependências entre os atores da organização. Cada relacionamento tem um dependente (**depender**) que depende de um outro ator (**dependee**) para alguma coisa (**dependum**). Caso o elemento que caracteriza a dependência (**dependum**) não esteja ao alcance do **dependee**, o **depender** sofrerá as implicações [3].

Modelo de Raciocínio Estratégica (SR): ao expandir um ator do modelo SD, que esteja em uma relação de dependência com outro ator, é observado um

conjunto de elementos (**goal**, **task**, **resource** e **softgoal**) relacionando-se com a propósito de apresentar as motivações e intenções do ator. Estas motivações do ator são sobre adotar uma ou outra determinada configuração para atender uma demanda. Os principais tipos de relacionamentos são representados pelas associações **means-end** e **task-decomposition** [3].

2.1 Limitações do Framework IStar

Há algumas limitações relacionadas a abordagem do framework iStar. Os problemas mais relevantes para a comunidade são apresentados na avaliação empírica apresentada no [10] que constatou que o Framework não tem as seguintes características: modularidade e escalabilidade. A Figura 1 ilustra um diagrama dos modelos do iStar que exige mais da representação gráfica devido a grande quantidade de elementos em uma mesma situação.

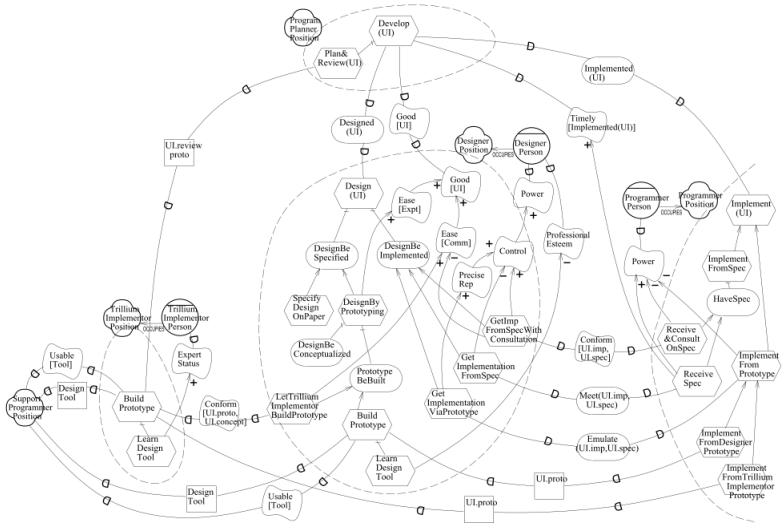

Figura 1. Exemplo do Modelo SR. Fonte: [3]

Escalabilidade: de acordo com [10], escalabilidade é o grau em que o framework de modelagem pode ser utilizado para lidar com aplicações de tamanhos diferentes. Assim como, de medir a capacidade de incluir novos elementos na modelagem sem alterar a compreensibilidade dos modelos.

Para o framework IStar, apesar de ter trabalhos que abordam esse problema, obter modelos escaláveis ainda é um desafio [4]. As soluções propostas, até o momento, não foram aceitas amplamente pela comunidade ou suficientemente abrangente para todas as instâncias do iStar. Fundamentado no [13], um mapeamento dos trabalhos que abordam escalabilidade do framework iStar, esta

proposta extraiu cinco características que enriquecem a definição de escalabilidade para o IStar, são elas: (i) a possibilidade de manipular vários atores em uma análise, sem impactar a modelagem em questão, (ii) tratar e manipular aplicações de tamanhos diferentes da mesma forma (iii) incorporar novos recursos sem grandes impactos, (iv) ter modelos em diferentes níveis de abstração, (v) ser facilmente modificável.

Modularidade: é a capacidade da linguagem de modelagem oferecer blocos de construção bem definidos para o modelo de construção. Os blocos de construção devem permitir o encapsulamento de estruturas internas do modelo em uma construção de modelagem real. Esta característica garante que as alterações em uma parte do modelo não serão propagadas para outras partes [10].

Esta capacidade é um outro desafio para o Framework. Pesquisadores enfrentam este desafio, bem com propõem alternativas [6,7,8,9,10]. Dos desafios mencionados, o trabalho de [6] coloca a questão da modularidade dos modelos como sendo o mais importante, e ratifica esta afirmação ao apresentar o reuso e a escalabilidade como consequências da modularidade. Portanto uma solução para modularidade pode trazer impactos positivos para reuso e escalabilidade.

3 Notação Textual do Framework iStar

Nesta seção, são explanados os requisitos que nortearam as decisões de projeto. Estas decisões foram extraídas a partir das dificuldades mencionadas em situações reais no [10]. Logo em seguida, a notação textual é apresentada através da definição gramatical e exemplos com o intuito de explanar as palavras reservadas e sua estrutura.

3.1 Requisitos da Notação Textual

Os **REQ 01** e **REQ 02** estão relacionados com a falta de modularidade e escalabilidade constatada pela avaliação empírica de [10] nos diagramas monolíticos, ou melhor, diagrama que contempla os dois modelos de forma única. Os **REQ 03** e **REQ 04** surgiram com base nas experiências e dificuldades relatadas na elaboração de notações textuais em [17,18,19,20].

REQ 01 - Uma modelagem composta por blocos inter-relacionada. Tornar possível que a abordagem da modelagem monolítica do framework iStar, possa também ser compreendida e visualizada como um conjunto de blocos separados, que são inter-relacionados. Com a possibilidade de construir blocos textuais inter-relacionados, pretende-se propiciar uma visualização gráfica dos blocos que forem importantes para o instante.

REQ 02 - Uma alternativa complementar aos modelos gráficos. Embora exista uma hegemonia da modelagem gráfica para abstração da realidade estudada, na Engenharia de Software são vistas outras alternativas para modelagem

de requisitos, como por exemplo: linguagem natural, especificação matemática e linguagem estruturada de projeto [21].

REQ 03 - Legível e compreensível para humanos. É desejado um grau de entendimento para qualquer leitor humano envolvido em uma modelagem que tenha como abordagem o framework iStar. A leitura da modelagem deve ser o mais próximo possível de uma leitura natural em comparação com o que é legível por máquina. Com isso, ambiciona-se compreensão e aprendizado da modelagem com a simples leitura do modelo textual (palavras e sentenças), sem necessitar gerar grande esforço para os envolvidos (engenheiro de requisito, engenheiro de negócio, desenvolvedores, usuários finais, dentre outros), ou necessidade de treinamentos para obter um nível de entendimento do framework para os envolvidos.

REQ 04 - Estrutura concisa e legível para máquinas. Ao mesmo tempo que seja compreensível por humano, almeja-se que uma máquina consiga analisar e compreender os padrões pré-estabelecidos de uma estrutura concisa. Esta característica tem a finalidade de possibilitar mecanismos controláveis e automatizáveis para conceder análises qualitativa (ex.: definir e detectar padrões de mal uso para a modelagem) e quantitativamente (ex.: oferecer informações exatas sobre a quantidade de vezes que um determinado elemento está sendo requisitado) da modelagem em questão.

3.2 Estrutura da Notação Textual

Dentre os requisitos supracitados, o requisito que direciona a modelagem (**todo**) como sendo uma composição de blocos (**partes**) inter-relacionados é o mais incisivo e determinante para definir a estrutura desta notação textual. Esta relevância do **REQ 01**, deve-se ao fato de que uma modelagem com bloco de construção está diretamente relacionado ao conceito de modularidade e por consequência, escalabilidade [10,6].

Diferente dos modelos orientado a metas do framework iStar, os modelos orientado a processo tem uma concepção modular naturalmente obtida através dos processo/subprocessos, por exemplo: Fluxogramas, Diagramas de Atividade UML, EPCs, BPMN e YAWL [22]. Esta concepção modular fornece o bloco de construção aos modelos orientados a processo, ou seja, um modelo (**todo**) orientado a processo é composto por processos/subprocessos (bloco de construção/**parte**).

O trabalho [11] indica que através da modularidade dos subprocessos, os modelos orientado a processos permitem vantagens quanto a escalabilidade e, além disso, conceitua um sistema modular como sendo uma composição de unidades (ou módulos) que são projetadas independentemente, mas ainda funcionam como um todo.

Diante da necessidade de uma concepção modular para os modelos do Framework iStar, este trabalho buscou nos conceitos e elementos empregados nos modelos SD e SR uma abstração (**parte**) que desacoplasse da modelagem (**todo**)

e pudesse ser naturalmente lida, interpretada e compreendida isoladamente das outras partes como nos modelos de orientado a processo.

Diferente de outras abordagens [5,7,9,10], a proposta traz uma sutil abstração dos próprios conceitos e elementos dos modelos do Framework. A sutil abstração (bloco de construção/**parte**) é a ***perspectiva do ator*** dentro de uma modelagem social (**todo**), ou seja, a complexidade social do ator dentro de uma organização.

Esta abstração, entendida como ***perspectiva do ator***, é estruturada da seguinte forma: (i) definição do ator e as suas associações organizacionais com outros atores, (ii) declarações das dependências externas na perspectiva **depender** e **dependee** e, por último, (iii) as definições de como o ator utiliza os elementos intencionais para atender alguma necessidade.

```
istarModels
    : perspectiveUnit+
    ;

perspectiveUnit
    :   organizationalStruture              // (i)    (a) as áreas (i) e (ii) foram concebidas do modelo Strategic Dependency
            externalRelationships           // (ii)
                internalRelationships       // (iii)  (b) a área (iii) abstraida do modelo Strategic Rationale

                EOF
    ;
```

Figura 2. Relação da estrutura da ***perspectiva do ator*** com os modelos do Framework iStar.

A Figura 2 apresenta a parte inicial da gramática com o objetivo de reforçar o conceito de uma modelagem social composta por um conjunto de ***perspectivas de atores*** (complexidade social do ator), nestas perspectivas estão diluídas as informações dos modelos SD (a) e SR (b) do Framework iStar de cada ator.

(i) definição do ator e as suas associações organizacionais com outros atores

A ***perspectiva do ator*** começa com a definição do tipo e da identificação do ator, cada perspectiva terá uma identificação única dentro da modelagem. Existem quatro tipos de atores, são eles: **actor**, **agent**, **role** e **position**. Após a declaração dos tipos do ator, deverá seguir para as declarações das associações com os outros atores, se houver. Existem seis tipos de associação entre os atores que, no modelo textual, assumem as seguintes palavras reservadas: **instanceOf**, **isPartOf**, **ocupies**, **covers**, **plays** e **isA**.

A Figura 3 apresenta a definição gramatical que especifica como deve ser iniciada uma ***perspectiva do ator***. Além da gramática, são mostrados exemplos da linguagem através de três perspectivas iniciadas. No cenário proposto por estas perspectivas, são extraídas algumas decisões de projeto. Por exemplo, a apresentação modular e hierárquica dos atores no modelo textual, diferente da compreensão e apresentação dos modelos gráficos existentes no framework iStar.

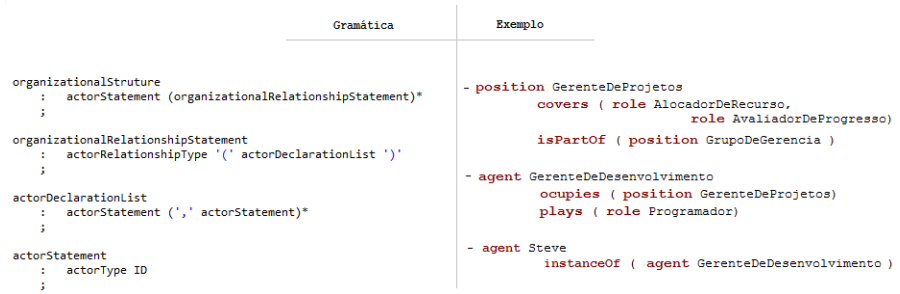

Figura 3. Gramática da definição organizacional do Framework Istar.

Uma apresentação através de ***perspectiva do ator*** significa a possibilidade de analisar a modelagem por partes, onde estas partes contém as informações encontradas nos modelos SD e SR a partir da perspectiva interessada. A concepção hierárquica para o modelo proposto, é a viabilidade de concentrar as definições e as declarações em cada perspectiva, sendo absorvidas pelas associações hierárquicas. Na Figura 3, a perspectiva do ***agent*** *Steve* absorve informações que deverão ser analisadas e compreendidas apenas na perspectiva do ***agent*** *GerenteDeDesenvolvimento* através da associação **instanceOf**.

(ii) declarações das dependências externas na perspectiva depender e dependee

Esta parte da ***perspectiva de ator*** está reservada para as declarações das relações externas, onde é possível compreender as dependências de um determinado ator. No modelo textual, estas declarações são evidenciadas sobre duas perspectivas, quando o ator assume a perspectiva do **depender** e/ou a do **dependee**. Embora o modelo textual possa estar evidenciando estas perspectivas, estas sempre existiram na semântica da notação gráfica.

Para não perder nenhuma compreensibilidade da modelagem, foi necessária a extração das sutilezas semânticas encontradas nos modelos gráficos e diluí-las em cada ***perspectiva do ator*** que comporá a modelagem. Estas sutilezas semânticas trazem uma redundância para o modelo textual, pois quando é observada a relação do ponto de vista do relacionamento de dependência, esta é informada na perspectiva do **depender** e do **dependee**.

No exemplo da Figura 4, juntamente com a definição gramatical, existe um cenário que demonstra os relacionamentos de dependência entre dois atores (*ProjetisDeIU* e *RevisorDeIU*) dentro de um ambiente de desenvolvimento de software. O *ProjetisDeIU* depende do *RevisorDeIU* para alcançar obter um Projeto Aprovado, enquanto o *RevisorDeIU* depende do *ProjetisDeIU* para realizar as Modificações Necessárias do Projeto.

A Figura 5 traz uma compreensão de como a proposta interpreta o relacionamento de dependência externa e as responsabilidades do **dependee** de acordo

Figura 4. Definição Gramatical dos Relacionamentos de Dependência.

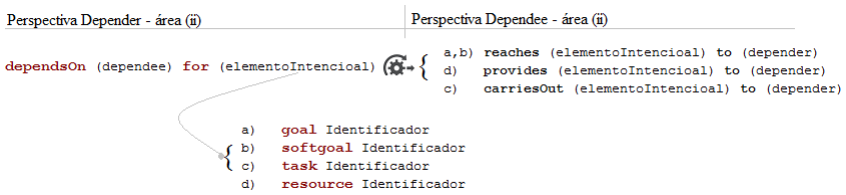

Figura 5. Relação da Estrutura Textual do Dependee com o tipo de Elemento Intencional utilizado no Relacionamento de Dependência.

com o elemento intencional utilizado. Ao expressar o relacionamento externo nas duas perspectivas, não se espera omitir a origem deste relacionamento. Pelo contrário, a sintaxe do **depender** segue um padrão no qual determina a origem da relação. No **dependee**, as estruturas e palavras são flexíveis a semântica dos relacionamentos de acordo com o elemento intencional contido no relacionamento de dependência.

Continuando na Figura 5, é visualizada uma engrenagem que representa o processo de verificação de integridade (interpretador). Este processo pretende diminuir o impacto negativo da redundância, pois as frases e palavras na perspectiva do **dependee** são reflexos da perspectiva **depender**. Esta relação permite que o processo de integridade consiga capturar e resolver as possíveis inconformidades.

(iii) as definições de como o ator utiliza os elementos intencionais para atender alguma necessidade

A última área (iii) está reservada para as declarações e definições dos elementos intencionais internos do ator. Com estas definições são especificada os relacionamentos entre os elementos internos para atender a uma demanda. No modelo gráfico estas definições textuais são encapsuladas dentro do ator, na qual outros atores da modelagem não têm acesso.

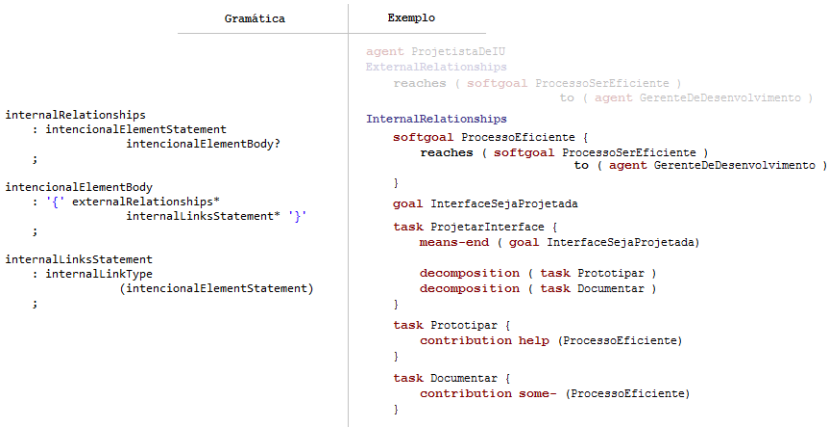

Figura 6. Definição Gramatical dos Elementos Internos do Ator com Exemplo.

Seguindo o cenário proposto na seção anterior, a Figura 6 apresenta as relações internas do *Projetista*, que assume a perspectiva de **dependee** no relacionamento externo com o *Gerente de Desenvolvimento* (**depender**). O *Projetista* é composto pelos elementos internos que se relacionam e indicam como pretendem atender a um **goal** (interface seja projetada) e **softgoal** (processo seja eficiente). Estes elementos internos são: (a) projetar uma interface gráfica do usuário que é refinada em, (b) realizar prototipação de interfaces para obter rápidas respostas sobre o andamento da interface e (c) documentar a escolha dos elementos gráficos utilizados.

4 Aplicação da Notação Textual

A Figura 7 de [23] apresenta um cenário que será utilizado para demonstrar a notação textual e fortalecer o conceito de uma modelagem social composta pela complexidade social dos atores (***perspectivas de atores***). O diagrama da Figura 7 retrata os dois modelos do Framework para abstrair um caso de gerenciamento de plano de saúde, onde o médico precisa da aprovação prévia para realizar o atendimento de um paciente.

O diagrama, apresentado na Figura 7, está contemplando as intenções de três atores com suas respectivas motivações e atividades. É importante ressaltar, como pode ser visto nas Figuras 8 e 9, que a ***perspectiva do ator*** compreende um bloco de construção (**parte**) do diagrama (**todo**). Neste bloco de construção, as informações contidas na perspectiva do *Paciente* são suficientes para gerar a representação gráfica da perspectiva em questão, como observado na Figura 8. A partir dos blocos de construções textuais da modelagem, podem, de acordo com a necessidade do analista, gerar representações gráficas de uma situação particular contida em um contexto maior, pois os blocos textuais poderão ser compreendido como uma base de conhecimento sobre a modelagem.

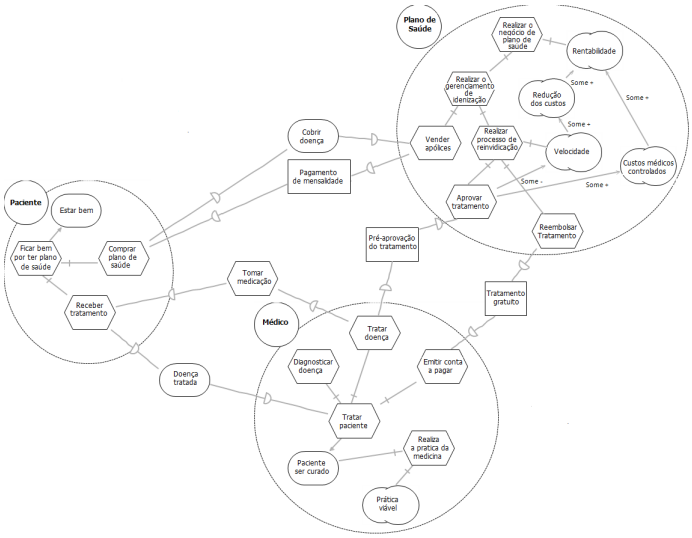

Figura 7. Diagrama dos modelos iStar para o exemplo do plano de saúde. Fonte: [23]

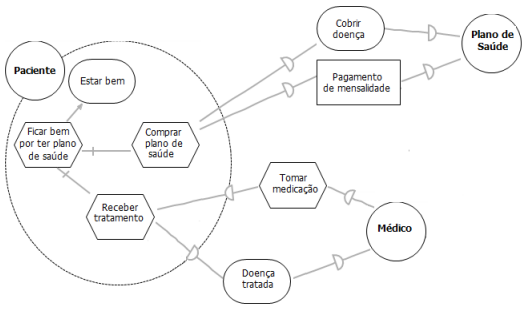

Figura 8. Abstração Gráfica da Perspectiva do Paciente.

```
actor  Paciente

ExternalRelationships
    dependsOn ( actor PlanoDeSaude) for ( goal cobrirDoencas )
    dependsOn ( actor Medico) for ( task doencaTratada)

    carriesOut ( task tomarMedicacao ) to ( actor Medico)
    provides ( resource pagamentoDaMensalidade ) to ( actor PlanoDeSaude)

ExternalRelationships
    softgoal estarBem

    task ficarBemPorTerPlanoDeSaude {
        decomposition ( task  comprarDoPlanoDeSaude)
        decomposition ( task receberTratamento)

        means-end ( goal estarBem)
    }

    task comprarDoPlanoDeSaude {
        dependsOn ( actor PlanoDeSaude) for ( goal cobrirDoencas )
        provides ( resource pagamentoDaMensalidade ) to ( actor PlanoDeSaude)
    }

    task receberTratamento {
        dependsOn ( actor Medico) for ( task doencaTratada)
        carriesOut ( task tomarMedicacao ) to ( actor Medico)
    }
```

Figura 9. Abstração textual da Perspectiva do Paciente.

Juntamente com a perspectiva do Paciente, existem mais duas perspectivas que compõem toda a modelagem social, as perspectivas são demonstradas na Figura 10. Nestas perspectivas, observa-se um grau de independência entre os blocos para realizar leitura/compreensão de uma parte da modelagem. Por exemplo, na linha 31 da perspectiva *Plano De Saúde* está a definição da tarefa Vender Apólice. Realizando a leitura do modelo textual é compreensível que deverá alcançar o objetivo Cobrir Doenças do Paciente para conseguir realizar a tarefa.

```
actor Medico

ExternalRelationships
    dependsOn ( actor PlanoDeSaude )
                for ( resource preAprovacaoDoTratamento)
    dependsOn ( actor Paciente)
                for ( resource tomarMedicacao)

    reaches ( goal doencaTratada) to ( actor Paciente)
    provides ( resource tratamentoGratuito)
                to ( actor PlanoDeSaude)

InternalRelationships


    task realizarPraticaDaMedicina {
        decomposition ( goal pacienteSerCurado)
        decomposition ( softgoal praticaViavel )
    }

    goal pacienteSerCurado

    softgoal realizarPraticaDaMedicina

    task (tratarPaciente) {
        reaches ( goal doencaTratada) to ( actor Paciente)

        means-end ( goal pacienteSerCurado)

        decomposition ( task tratarDoenca )
        decomposition ( task diagnosticarDoenca)
        decomposition ( task emitirContaAPagar)
    }

    task diagnosticarDoenca

    task (tratarDoenca) {
        dependsOn ( actor PlanoDeSaude )
                    for ( resource preAprovacaoDoTratamento)
        dependsOn ( actor Paciente)
                    for ( resource tomarMedicacao)
    }

    task emitirContaAPagar {
        provides ( resource tratamentoGratuito)
                    to ( actor PlanoDeSaude)
    }
```

```
actor PlanoDeSaude

ExternalRelationships
    dependsOn ( actor Paciente)
                for ( resource pagamentoDaMensalidade)
    dependsOn ( actor Medico)
                for ( resource tratamentoGratuito)

    reaches ( goal cobrirDoencas)
                to ( actor Paciente)
    provides ( resource preAprovacaoDoTratamento)
                to ( actor Medico)

InternalRelationships

    task realizarNegocioDePS {
        decomposition ( task realizarGerenciamentoDeIdenizacao)
        decomposition ( softgoal rentabilidade)
    }

    softgoal rentabilidade

    task realizarGerenciamentoDeIdenizacao{
        decomposition ( task venderApolice)
        decomposition ( task realizarProcessoDeReinvidicacao)
    }

    task venderApolice{
        reaches ( goal cobrirDoencas)
                    to ( actor Paciente)
    dependsOn ( actor Paciente)
                for ( resource pagamentoDaMensalidade)
    }

    task realizarProcessoDeReinvidicacao{
        decomposition ( task aprovarTratamento)
        decomposition ( task reembolsarTratamento)
        decomposition ( softgoal velocidade )
    }

    softgoal velocidade

    task aprovarTratamento {
        provides ( resource preAprovacaoDoTratamento)
                    to ( actor Medico)

        contribution some- ( softgoal velocidade)
    }
```

Figura 10. Abstração textual das Perspectivas do Medico e Plano de Saúde.

5 Trabalhos Relacionados e Discussões

Os avanços tecnológicos das ferramentas, que lidam diretamentamente com modelos textuais, oferecem benefícios como: uma maior facilidade para integração com outros modelos, maior facilidade na manipulação das linguagens, qualidade na formatação, versionamento e independência de plataforma. Estes benefícios são apresentados em trabalhos como [17,18,19,20,24,25]. Diante desses estudos, que mostram a relevância dos modelos textuais, [24] já afirma que uma abordagem híbrida pode ser uma necessidade de projeto com o intuito de aproveitamento das melhores práticas de ambos estilos. Atualmente, há três tipos de abordagens para modelagem em geral: textual, gráfica e híbrida (gráfica e textual) [25].

Além dos benefícios dos modelos textuais, esta proposta oferece ao Framework iStar: (a) uma concepção natural de bloco de construção (perspectiva/parte/módulo), (b) um estilo de leitura para os relacionamentos entre os elementos e (c) um caminho para construção de uma abordagem híbrida. A concepção natural de bloco de construção é de vital importância devido a possibilidade de gerenciar a complexidade dos diagramas para modelos complexos. O conceito de ***perspectiva do ator*** está diretamente relacionado com o entendimento de modularidade para uma linguagem de modelagem que no trabalho [10] define como a capacidade de oferecer blocos de construção bem definidos para o modelo de construção.

5.1 Descrição dos Trabalhos Diretamente Relacionados

Nos artigos [19,20], é apresentado um modelo textual para uma Linguagem de Requisitos Orientado a Objetivo (TGRL) como uma forma de facilitar a leitura, manipulação e revelação de maus uso da notação.

A comunidade iStar está atenta aos modelos textuais e seus benefícios, trabalhos como [26,27], surgem com o objetivo de alcançar a interoperabilidade dentro de uma variação de modelos do framework iStar. iStarML [26] é uma linguagem de marcação que determina um formato para troca de dados baseado em XML para estabelecer uma efetiva comunicação entre a comunidade que se expande. iStarJSON [27] é modelo textual leve para troca de dados entre os modelos iStar, a justificativa desta proposta é uso da tecnologia que deixa menos verboso e mais flexível, fácil de programar e legível para humanos.

5.2 Comparação dos Trabalhos

A tabela 1 demonstra a relação desta proposta com os trabalhos relacionados mediante as características desejadas por cada notação textual. Ou seja, se as características desejada por esta proposta são as mesmas que conduziram outros modelos textuais para o Framework iStar.

- REQ 01 - Uma modelagem composta por blocos inter-relacionada;
- REQ 02 - Uma alternativa complementar aos modelos gráficos;
- REQ 03 - Legível e compreensível para humanos;
- REQ 04 - Estrutura concisa e legível para máquinas.

Projetos	REQ 01	REQ 02	REQ 03	REQ 04	Ano
TGRL [19]			x	x	2015
iStarML [26]				x	2008
iStarJSON [27]				x	2016
Esta Proposta	x	x	x	x	2016

Tabela 1. Tabela de comparação de trabalhos relacionados com a Proposta

6 Considerações finais

As notações gráficas, na maioria das vezes, mostram-se mais eficientes e intuitivas para análise e interpretação, mas não é possível gerenciar a complexidade dos modelos do framework iStar. A proposta apresentada é proporcionar um caminho para gerenciar esta complexidade dos modelos iStar.

A notação textual traz a possibilidade de termos uma modelagem (**todo**) através dos blocos de construções (***perspectiva de ator***, **parte** ou módulo). Estes blocos poderão proporcionar modularidade, como demonstrado na aplicação da notação textual, e dividir uma modelagem em partes menores independentes e gerenciáveis. Além disso, estas partes poderão ser compreendido como uma base de conhecimento para gerar diagramas de acordo com a necessidade particular de um contexto maior.

Para trabalhos futuros, além da demonstração do conjunto de ferramentas (editor, analisador qualitativo e quantitativo) que está em desenvolvimento, pretende-se construir um plugin do eclipse para apoiar uma abordagem híbrida, obtendo os benefícios de ambos os estilos. Conceitualmente, esta proposta se encaixa em uma definição escalável pela modularidade especificada pela ***perspectiva de ator***, mas uma avaliação mais elaborada com o intuito de responder o quanto esta solução pode ser compreendida pelos os usuários como modular, escalável e legível para qualquer envolvido no ambiente de desenvolvimento.

Referências

1. Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7):1645–1660, 2013.
2. Joseph Barjis. The importance of business process modeling in software systems design. *Science of Computer Programming*, 71(1):73–87, 2008.
3. Eric Siu-Kwong Yu. *Modelling Strategic Relationships for Process Reengineering*. PhD thesis, Toronto, Ont., Canada, Canada, 1996. UMI Order No. GAXNN-02887 (Canadian dissertation).
4. Eric S. Yu. Social modeling and i*. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5600 LNCS(c):99–121, 2009.
5. Carlos Miguel Nunes. Uma linguagem de domínio específico para a framework i*. Master's thesis, Universidade Nova de Lisboa, Lisboa, Portugal, 2009.
6. Xavier Franch. Incorporating modules into the i* framework. In *International Conference on Advanced Information Systems Engineering*, pages 439–454. Springer, 2010.
7. Xavier Franch. The i * framework: The way ahead (presentation). *Research Challenges in Information Science (RCIS), 2012 Sixth International Conference on*, (May):1–3, 2012.
8. F. Alencar, J. Castro, A. Moreira, J. Araujo, C. Silva, R. Ramos, and J. Mylopoulos. Integration of aspects with i* models. *AOIS 2006*, 2006.
9. Fernanda Alencar, Jaelson Castro, Marcia Lucena, Emanuel Santos, Carla Silva, Jo Araújo, and Ana Moreira. Towards modular i* models. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, pages 292–297, 2010.

10. Hugo Estrada Esquivel. *A service-oriented approach for the i * framework*. PhD thesis, Valencia University of Technolog, 2008.
11. Hajo Reijers and Jan Mendling. Modularity in process models: Review and effects. In *International Conference on Business Process Management*, pages 20–35. Springer, 2008.
12. David Harel and Bernhard Rumpe. Modeling Languages : Syntax , Semantics and All That Stu. pages 1–28, 2000.
13. Paulo Lima, Jéssyka Vilela, Enyo Goncalves, João Pimentel, Ana Holanda, Jaelson Castro, Fernanda Alencar, and Maria Lencastre. Scalability of iStar: a Systematic Mapping Study. *Wer*, 45:289–302, 2016.
14. Avner Ottensooser, Alan Fekete, Hajo A. Reijers, Jan Mendling, and Con Menictas. Making sense of business process descriptions: An experimental comparison of graphical and textual notations. *Journal of Systems and Software*, 85(3):596–606, 2012.
15. Luc Engelen and Mark van den Brand. Integrating textual and graphical modelling languages. *Electronic Notes in Theoretical Computer Science*, 253(7):105–120, 2010.
16. Fabio Penha, Marcia Lucena, Leonardo Lucena, Celso Angra, and Fernanda Alencar. A proposed textual model for i-star. *CEUR Workshop Proceedings*, 1674:7–12, 2016.
17. Quentin Boucher, Andreas Classen, Paul Faber, and Patrick Heymans. Introducing TVL, a Text-based Feature Modelling. *Fourth International Workshop on Variability Modelling of Software-Intensive Systems, Linz, Austria, January 27-29, 2010. Proceedings*, pages 159–162, 2010.
18. Martin Mazanec, Ondřej Macek, and Martin Mazanec. On General-purpose Textual Modeling Languages On General-purpose Textual Modeling Languages. pages 1–12, 2012.
19. Vahdat Abdelzad, Daniel Amyot, Sanaa A. Alwidian, and Timothy C. Lethbridge. A textual syntax with tool support for the goal-oriented requirement language. *CEUR Workshop Proceedings*, 1402(istar):61–66, 2015.
20. Joachim Fischer, Markus Scheidgen, Ina Schieferdecker, and Rick Reed. Adding a Textual Syntax to an Existing Graphical Modeling Language: Experience Report with GRL. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9369:159–174, 2015.
21. I. Sommerville. *Engenharia de software*. PEARSON BRASIL, 2011.
22. C Baldwin and K Clark. Managing in an Age of Modularity. *Harvard Business Review*, (september), 1997.
23. Celso Luiz Agra De Sa Filho. Transformação do modelo i* em user stories: razões, intenções e NFR na documentação ágil. Master's thesis, FUNDAÇÃO UNIVERSIDADE DE PERNAMBUCO, Pernambuco, Brasil, 2015.
24. Th Tse and L Pong. An examination of requirements specification languages. *The Computer Journal*, 34(4):143–152, 1991.
25. Hans Grönninger, Holger Krahn, Bernhard Rumpe, Martin Schindler, and Steven Völkel. Textbased modeling. *arXiv preprint arXiv:1409.6623*, 2014.
26. Carlos Cares, Xavier Franch, Anna Perini, and Angelo Susi. IStarML: An XML-based interchange format for i* models. *CEUR Workshop Proceedings*, 322:13–16, 2008.
27. Oscar Franco-Bedoya, David Ameller, Dolors Costal, and Lidia López. istarjson: A lightweight data-format for i* models. *CEUR Workshop Proceedings*, 2016.