

Requisitos – Aula 20

Professores: Milene Serrano e Maurício Serrano

Agenda

- › Considerações Iniciais
- › Modelagem de Requisitos – GORE
 - i* (iStar)- Intencionalidade Distribuída
- › Debate
- › Metodologias e/ou Processos de Desenvolvimento
- › Considerações Finais

Considerações Iniciais

Considerações Iniciais

Conhecemos propostas de abordagens de modelagem de requisitos mais tradicionais (ex. UML); ágeis (ex. artefatos do Scrum e SAFe); e, nessa aula, iremos conhecer uma abordagem mais emergente, chamada ^{i*}.

Conhecemos ainda uma abordagem mais nova:

GORE – Goal Oriented Requirements Engineering



Considerações Iniciais

Sabemos que se trata de uma proposta de Engenharia de Requisitos que se orienta pela especificação de metas.

My Goals

- 1.
- 2.
- 3.

Metas capturam, em diferentes níveis de abstração, os vários objetivos que o sistema em desenvolvimento precisa atingir.

Essa área têm despertado o interesse da comunidade da Engenharia de Requisitos nos últimos anos.

My Goals

- 1.
- 2.
- 3.

Considerações Iniciais

Vimos que é uma evolução em relação ao paradigma da Orientação a Objetos.

Procura manter a especificação de alternativas, possibilitando o gerenciamento de conflitos; separando informações mais estáveis das informações mais voláteis; e permitindo tratar os objetivos com base na identificação de requisitos que viabilizem esses objetivos.

Considerações Iniciais

Mas, essa tarefa de especificar as metas não é fácil.

My Goals

- 1.
- 2.
- 3.

Demandam aplicar várias técnicas de elicição, em especial, introspecção e observação.

Adicionalmente, precisam ficar claras algumas diferenças entre as principais abstrações dessa abordagem...

Considerações Iniciais

As metas podem ser:



Metas ou **Goals** ou **Hardgoals**: representam um objetivo que pode ou não ser atingido. Comporta-se como uma variável booleana.



Metas-flexíveis ou **Softgoals**: representam desejos que são relevantes para o sucesso do software, mas cuja elicitação não é tão simples, pois são subjetivos, abstratos, sendo difícil, por exemplo, dizer se foram ou não atingidos. Podem ter sido parcialmente atingidos. Podem ter sido 80% atingidos na opinião de alguns; mas 20% atingidos na opinião de outros...



Considerações Iniciais

- Pedrinho seja aprovado na disciplina de Requisitos de Software



Pode ser APROVADO, ou pode ser REPROVADO.

Considerações Iniciais



Reparam na voz passiva...

Apesar de não ser obrigatório o uso dessa construção na definição de uma meta, façam uso dessa forma para facilitar a especificação.

Subject (Substantive or Name) + Verb (Passive Voice) + Complement

Exemplos em Português:

- Ana seja matriculada na disciplina de Física I.
- Disciplina seja cumprida em duas horas.
- Armário seja pintado.

Basicamente:

```
IF (NOTA >= 5)
{
    APROVADO
}
ELSE
{
    REPROVADO
}
```

Exemplos em Inglês:

- Peter must be registered at UnB.
- The book must be read in full by Anny.
- The school must be opened tomorrow morning.

Considerações Iniciais



Aqui, é ligado a um critério qualitativo...

Procurem usar a construção:

Softgoal [topic]
Quality_Criteria [Substantive]

Exemplos em Português:

- Bom Desempenho [Aplicativo]
- Privacidade Adequada [Serviço disponível na Nuvem]

Se a disciplina é vista como fácil, ser aprovado com nota = 5, pode não ser um BOM RENDIMENTO.

Mas, se a disciplina é vista como muito difícil, ser aprovado com nota = 5, pode ser considerado um BOM RENDIMENTO.

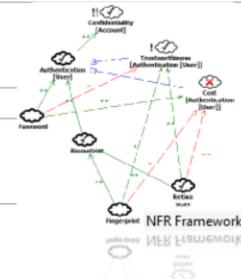
Notem como é relativo!
Não é tão simples dizer que teve BOM RENDIMENTO ou não teve BOM RENDIMENTO, sem relativizar, considerando outros parâmetros.

Exemplos em Inglês:

- Good Security [Software]
- Adequate Usability [Software Interface]

Considerações Iniciais

- Existem alguns frameworks conceituais que se orientam pela Engenharia de Requisitos Orientada à Meta. Tais como:
 - i* (iStar) ou Intencionalidade Distribuída. Veremos nessa aula. 
 - NFR Framework. Veremos mais adiante no curso.
 - Outros.



Modelagem de Requisitos

*i**

i*

http://istar.rwth-aachen.de/tiki-view_articles.php
<http://www.cs.toronto.edu/km/istar/>

Trata-se de um framework conceitual para condução da Engenharia de Requisitos orientada às propriedades intencionais dos envolvidos.

Entende-se por propriedades intencionais:

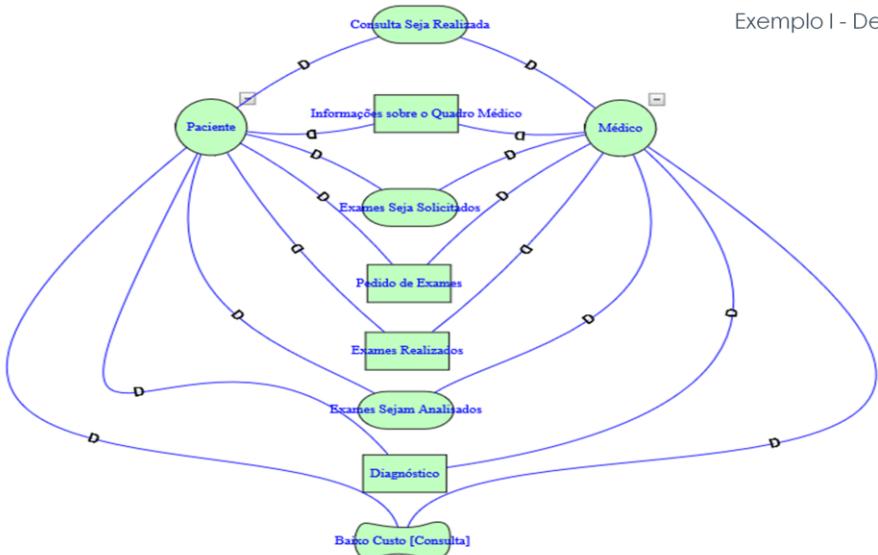
- Metas ou *Goals*;
- Metas-Flexíveis ou *Softgoals*, e
- outras abstrações.

Baseia-se em dois modelos:

- Modelo Estratégico de Dependência, ou *Strategic Dependency Model (SD)*, e
- Modelo de Raciocínio Estratégico, ou *Strategic Rationale Model (SR)*.

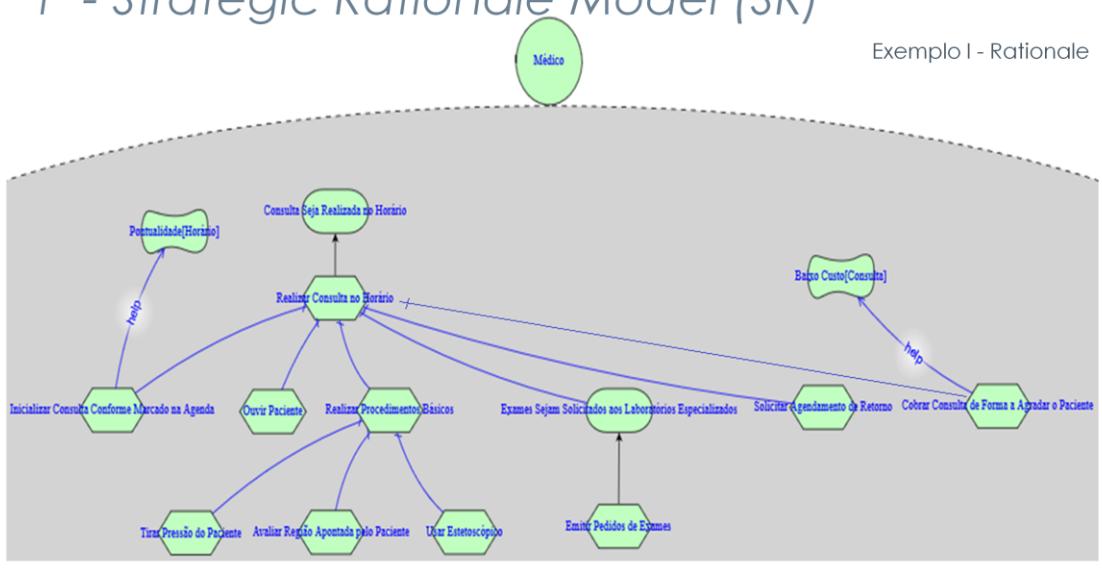
*i** - Strategic Dependency Model (SD)

Exemplo I - Dependência



i* - Strategic Rationale Model (SR)

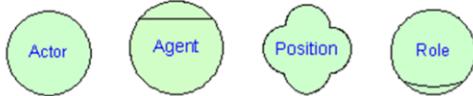
Exemplo I - Rationale



Reparem que se colapsarmos as fronteiras, teremos o SD.

i^* - ACTOR(s)

Actors



Ator

- Representa um ator comum; pessoa física ou jurídica, cujo rationale não é automatizado. Ex: João, Ana ou outra pessoa.

Agent

- Uma variação da abstração de ator, cujo rationale foi automatizado. Ex: Sistema.

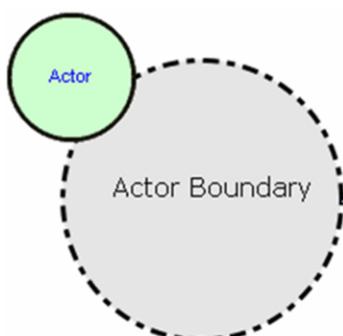
Position

- Uma variação da abstração de ator que representa uma posição dentro do contexto. Ex: na FGA, os cargos de Diretor ou de Coordenador de curso são posições. Posições acumulam papéis.

Role

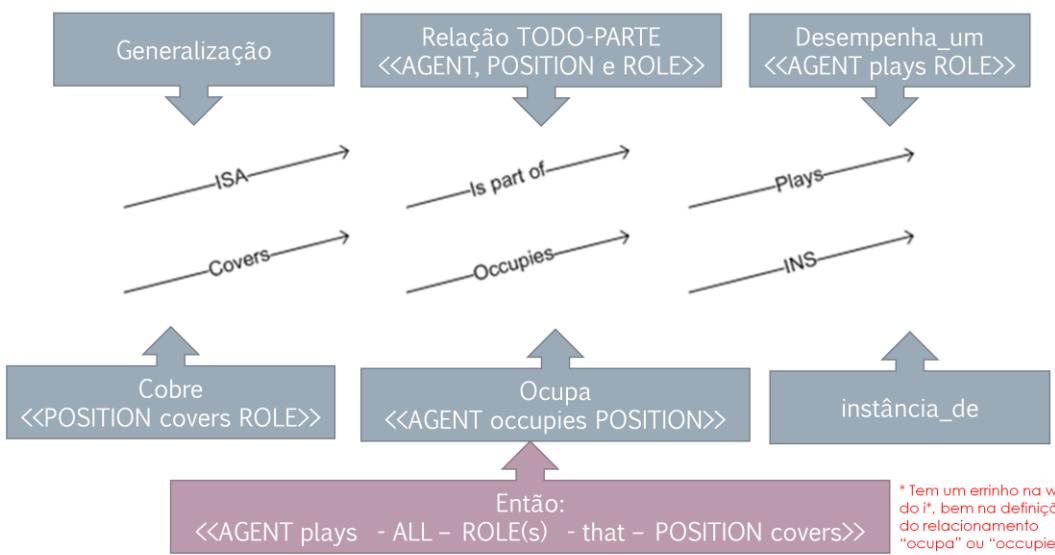
- Outra variação da abstração de ator usada quando se deseja representar um papel desempenhado dentro do contexto. Ex: no curso de Engenharia de Software da FGA, alguém, na posição de Coordenador, desempenha os papéis de principal representante do corpo docente de Engenharia de Software; mediador nas reuniões, e responsável pela elaboração da lista de ofertas do curso de Engenharia de Software.

i^* - Fronteira

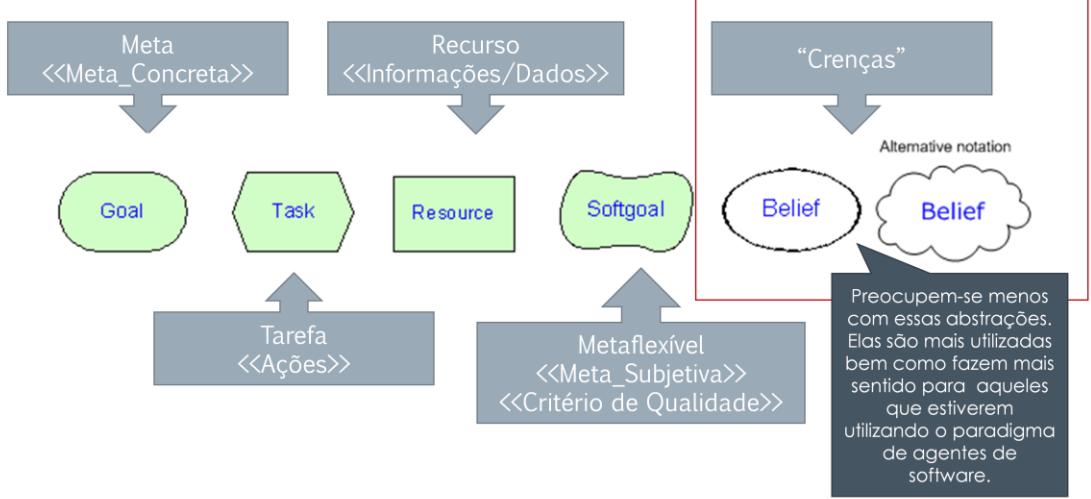


Área de fronteira do ator,
limitando o escopo de sua
atuação e, portanto, da
representação de seu
rationale.

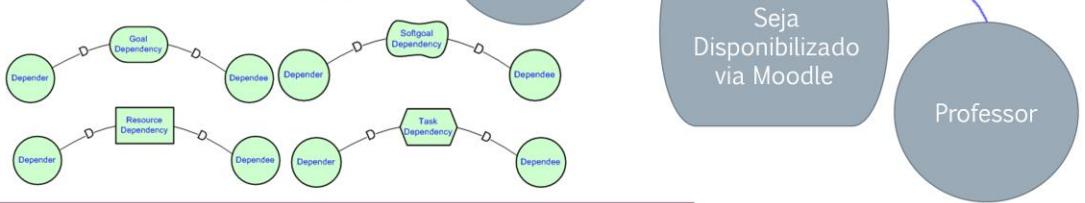
*i** - Associações entre ACTOR(s)



*i** - Abstrações Complementares



*i** - Links Gerais

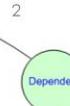


Dependência
⟨entre as variações da abstração de ACTOR⟩



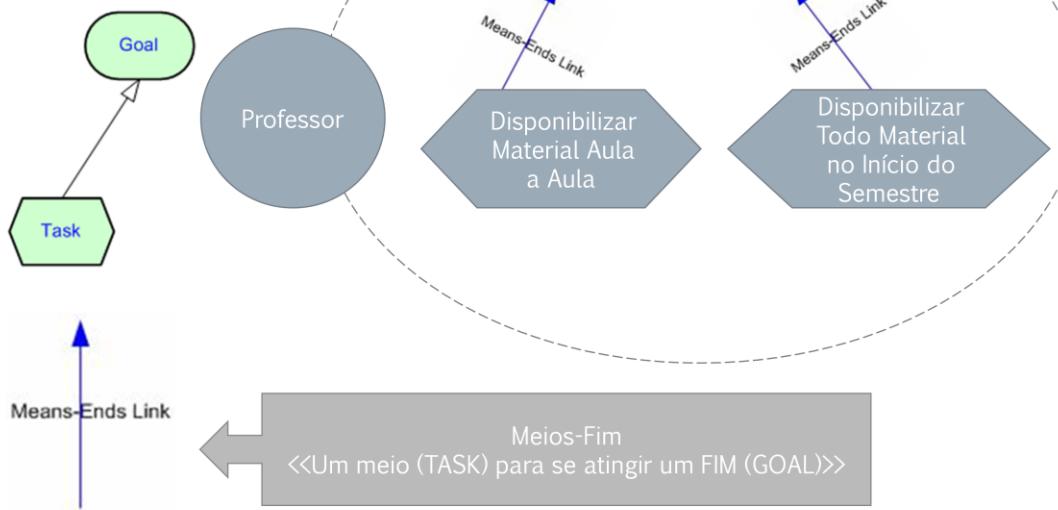
1. Alguém depende que a meta seja atingida, mas não tem alguém para delegar.

One-Side Dependency

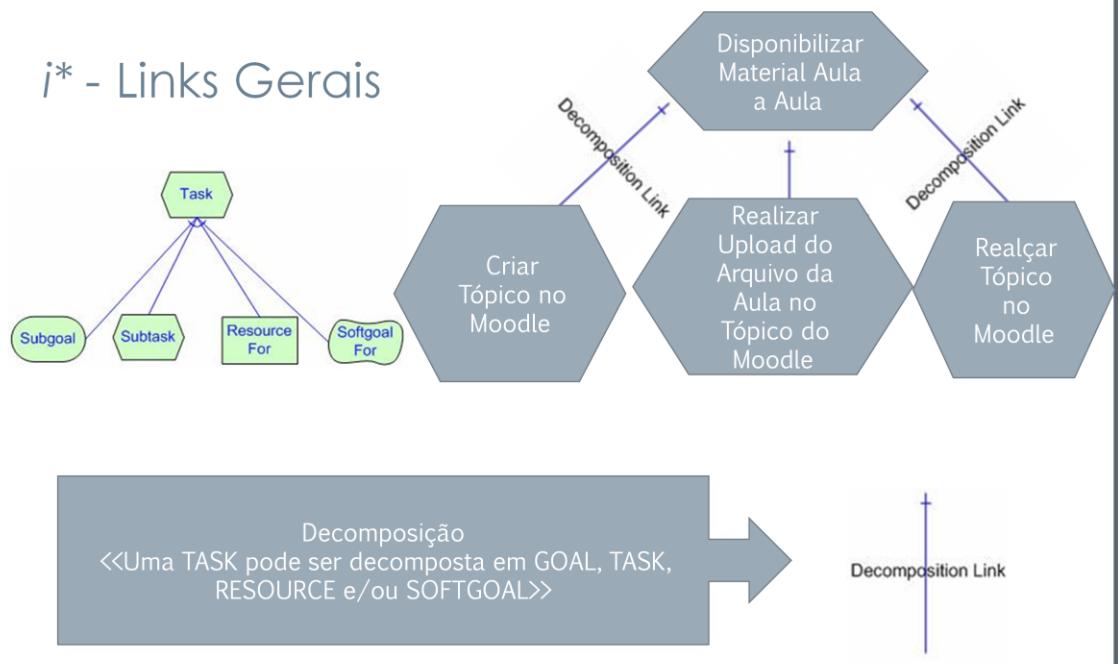


2. Alguém pode assumir a demanda de atingir a meta, mas não tem alguém delegando.

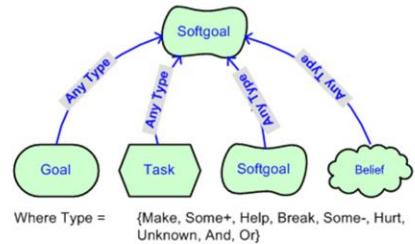
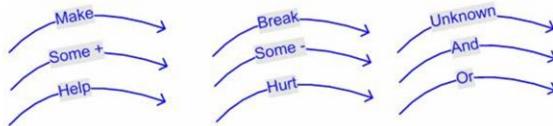
i* - Links Gerais



i* - Links Gerais

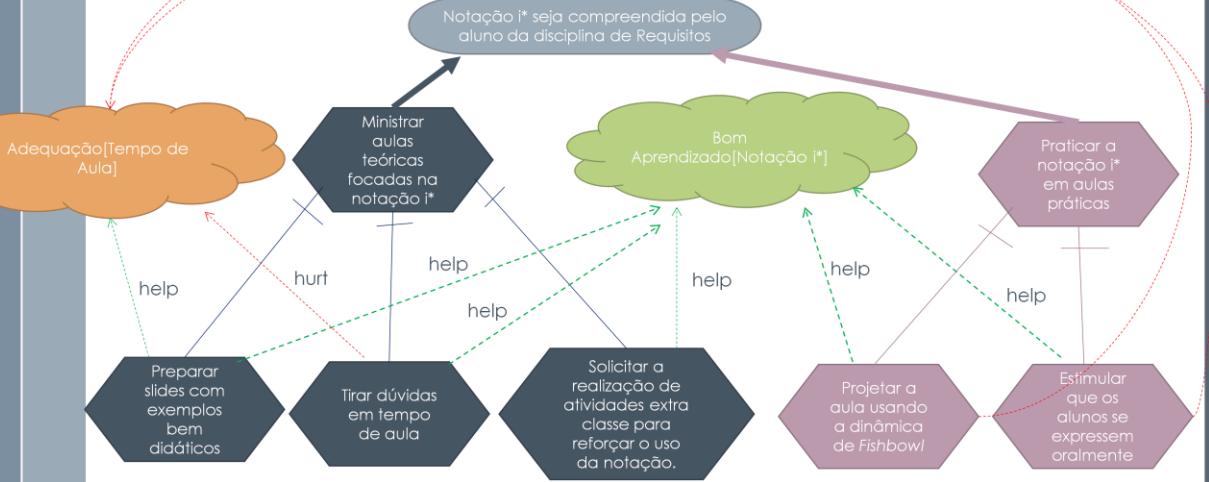


i* - Links de Contribuição

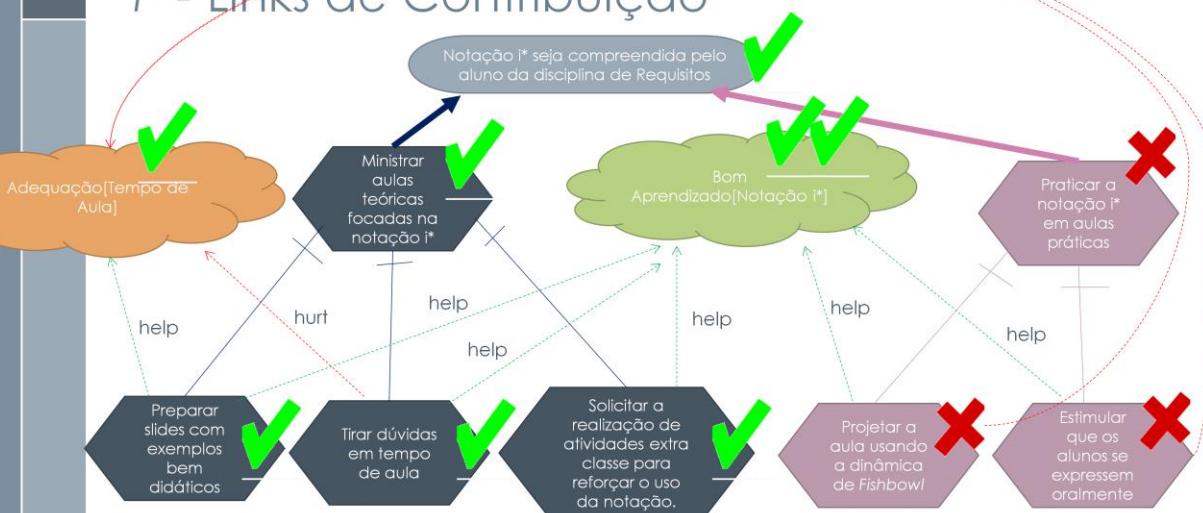


Link	Significado
Make	Contribuição tão positiva a ponto de satisfazer "algo" sob a perspectiva dos envolvidos.
Help	Contribuição positiva parcial, que sozinha não chega a satisfazer "algo" sob a perspectiva dos envolvidos.
Unknown	Contribuição, cuja polaridade é desconhecida.
Hurt	Contribuição negativa parcial, que sozinha não chega a negar/quebrar "algo" sob a perspectiva dos envolvidos.
Break	Contribuição tão negativa a ponto de negar/quebrar "algo" sob a perspectiva dos envolvidos.
Some +	Contribuição positiva, cuja intensidade não se pode determinar.
Some -	Contribuição negativa, cuja intensidade não se pode determinar.
And	"Pai" é satisfeito se_somente_se todos os "filhos" forem satisfeitos sob a perspectiva dos envolvidos.
Or	"Pai" é satisfeito se_somente_se um dos "filhos" é satisfeito sob a perspectiva dos envolvidos.

*i** - Links de Contribuição



*i** - Links de Contribuição



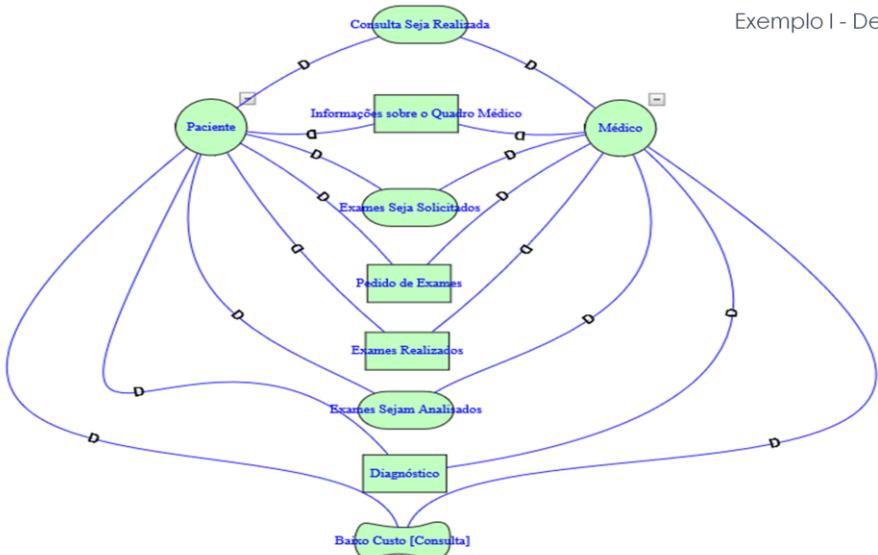
:

Em um segundo momento....

Variabilidade

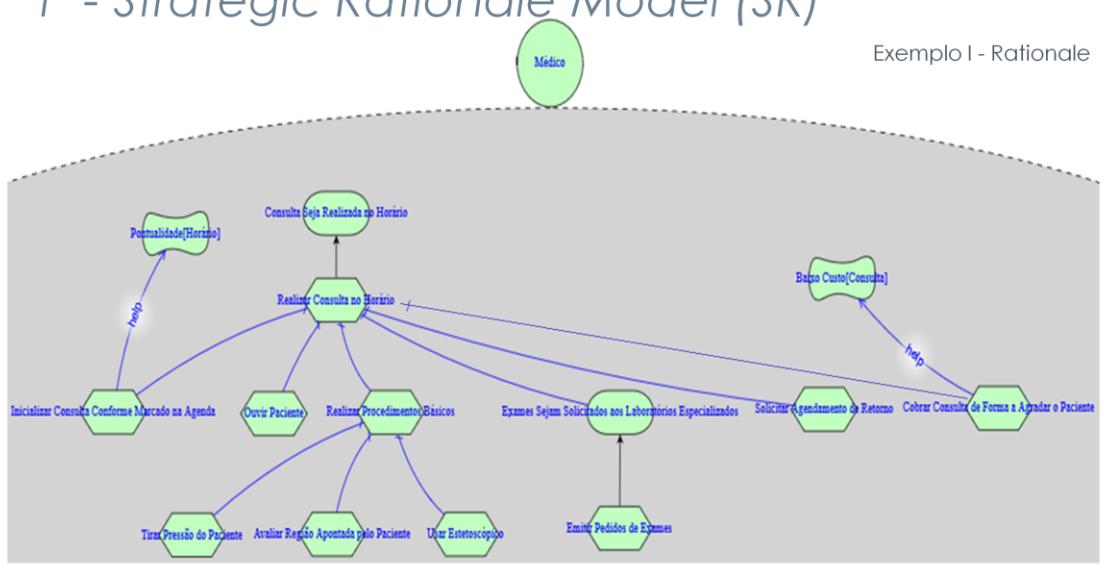
*i** - Strategic Dependency Model (SD)

Exemplo I - Dependência



i* - Strategic Rationale Model (SR)

Exemplo I - Rationale

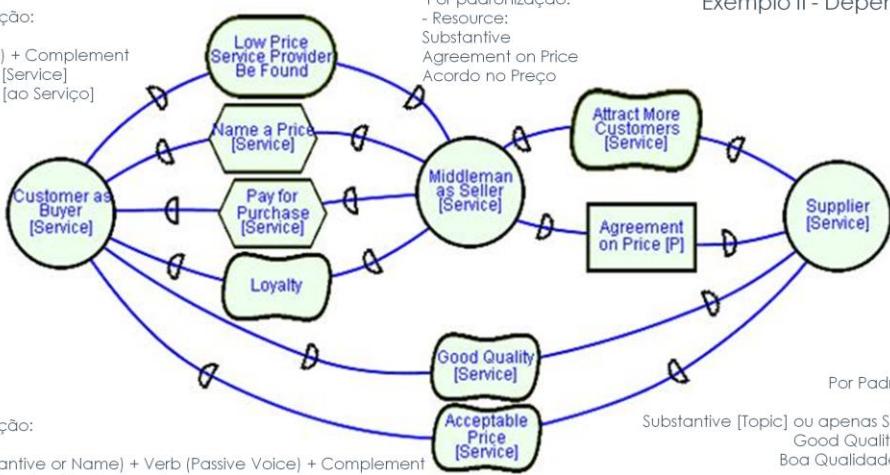


Reparem que se colapsarmos as fronteiras, teremos o SD.

i* - Strategic Dependency Model (SD)

Por padronização:

- Task:
Verb (Infinitive) + Complement
Name a Price [Service]
Dar um Preço [ao Serviço]



Por padronização:

- Goal:
Subject (Substantive or Name) + Verb (Passive Voice) + Complement
Low Price Service Provider Be Found
Provedor de Baixo Preço Seja Encontrado

Por padronização:

- Resource:
Substantive
Agreement on Price
Acordo no Preço

Exemplo II - Dependência

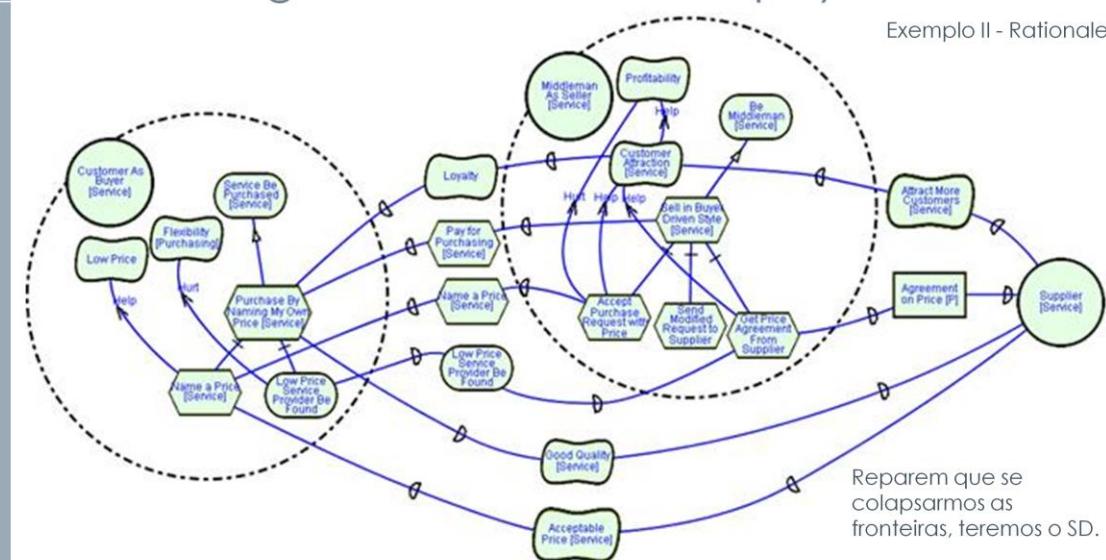
Por Padronização:
- Softgoal:
Substantive [Topic] ou apenas Substantive

Good Quality [Service]
Boa Qualidade [Serviço]

Loyalty
Lealdade

i* - Strategic Rationale Model (SR)

Exemplo II - Rationale



i* - Ferramentas

Temos uma que está sendo desenvolvida aqui na FGA, em um trabalho de TCC.
Mas, a mesma ainda está em fase de construção... :(

Existem várias ferramentas, conforme pode ser visto acessando o link apresentado no rodapé do slide.

A OpenOME é uma das mais recomendadas.

Código aberto.

Bem completa.

Feita pela própria comunidade do i*, na University of Toronto (Uoft).

OpenOME

<http://istar.rwth-aachen.de/tiki-index.php?page=i%2A+Tools>

<https://se.cs.toronto.edu/trac/ome>

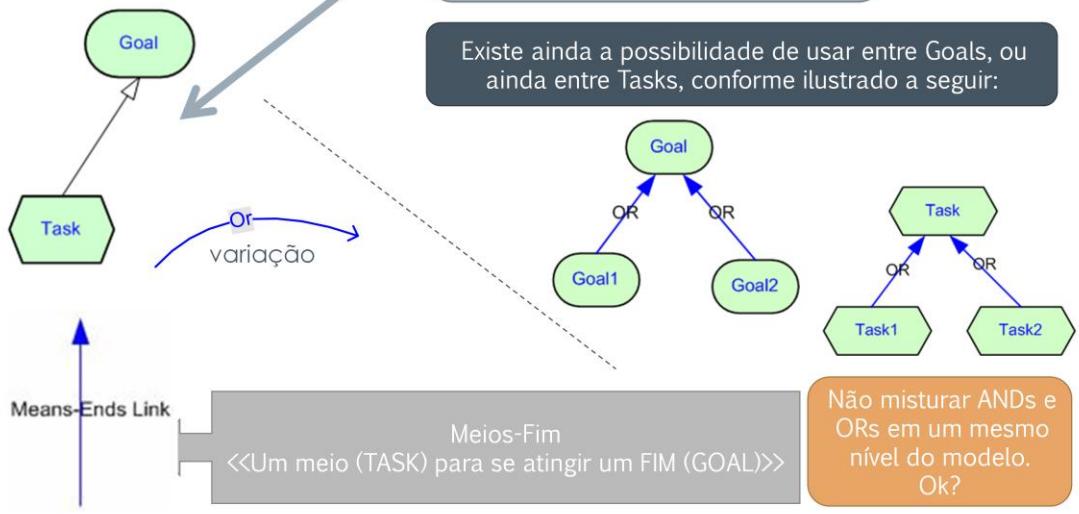
<https://se.cs.toronto.edu/trac/ome/wiki/GettingStarted>

Algumas Mudanças nas Versões mais Recentes da Notação i*

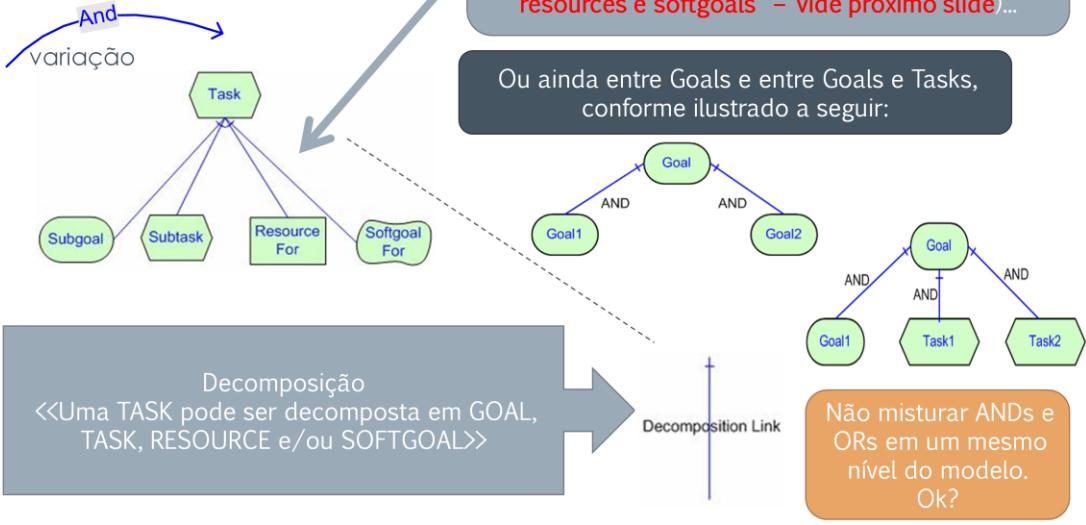
Simplificações

Entendido, nas últimas versões da notação, como um OR, que pode ser utilizado, por exemplo, como era na notação anterior...

Existe ainda a possibilidade de usar entre Goals, ou ainda entre Tasks, conforme ilustrado a seguir:

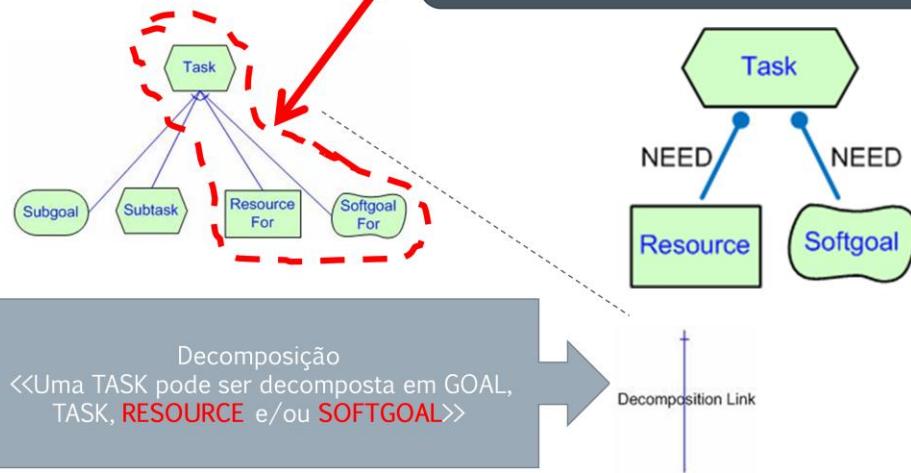


Simplificações



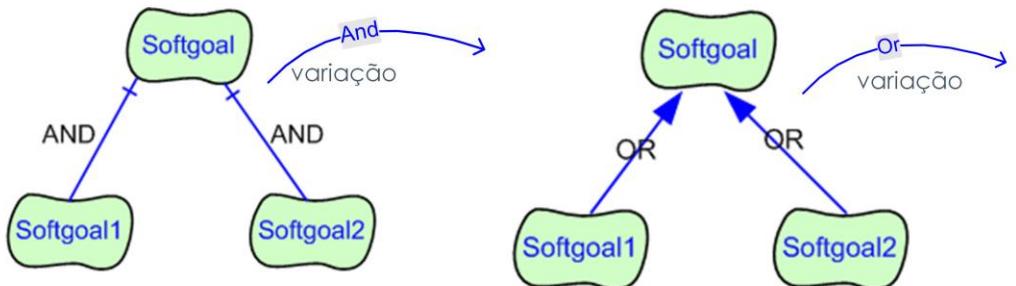
Simplificações

Entendido, nas últimas versões da notação, como um NEED, representado conforme ilustrado a seguir:



Simplificações

Também são possíveis...



ANDs e ORs entre softgoals, conforme visto na notação do NFR Framework.

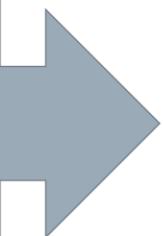
Debate

Debate

Trata-se de um *framework* conceitual para modelagem muito mais completo...

Mantem-se, inclusive, a rastreabilidade quanto às decisões de projeto, estabelecidas junto aos envolvidos.

Os “porquês”, os “comos”, as “alternativas”, os interessados, as fontes e outros detalhes ficam especificados...



Reparam que o foco é em ambos os requisitos, sejam:

FUNCIONAIS

Ou

NÃO-FUNCIONAIS

Debate

Esse tipo de enfoque, colabora com, por exemplo:



...



Adequação

Aprendizado



Adaptabilidade



Previsibilidade



Tomada de Decisão



Metodologias e/ou Processos de Desenvolvimento

Metodologias de Desenvolvimento

A notação do framework i* bem como outros artefatos de modelagem orientados à meta são mais divulgados na comunidade acadêmica.

Em termos de atuação no mercado, têm-se registros mais no exterior (em Países como Canadá), no qual as empresas e organizações valorizam muito iniciativas que apresentem erros e acertos no nível organizacional de uma empresa. Mostrando sobrecarga de setores e/ou de posições administrativas, dentre outras revelações que permitam a empresa se alinhar, ganhar posição estratégica...

Portanto, existem algumas abordagens, metodologias e/ou processos que conduzem o desenvolvimento de software via orientação à meta.

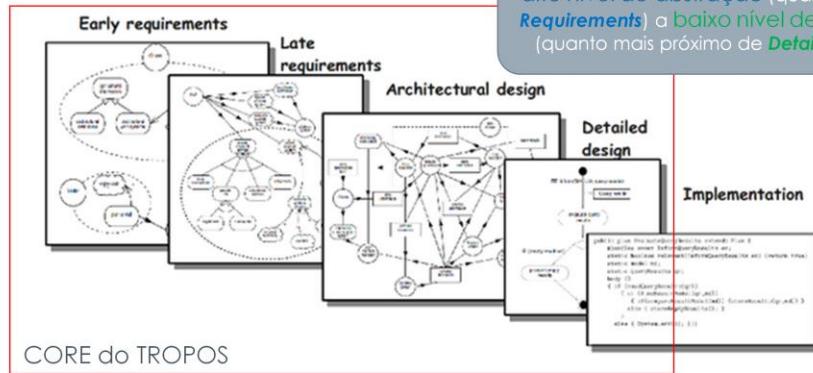


Um dos mais relevantes é o:

TROPOS, uma metodologia de desenvolvimento de software orientada à agentes, sendo esses agentes desenhados, especificando-se suas metas, tarefas, recursos, dependências e rationale. Disponível em:
<http://www.troposproject.eu/>

TROPOS

Fases do TROPOS:



Vantagens...

A vantagem que mais chama a atenção nesse tipo de desenvolvimento é permitir manter o rastro das decisões...



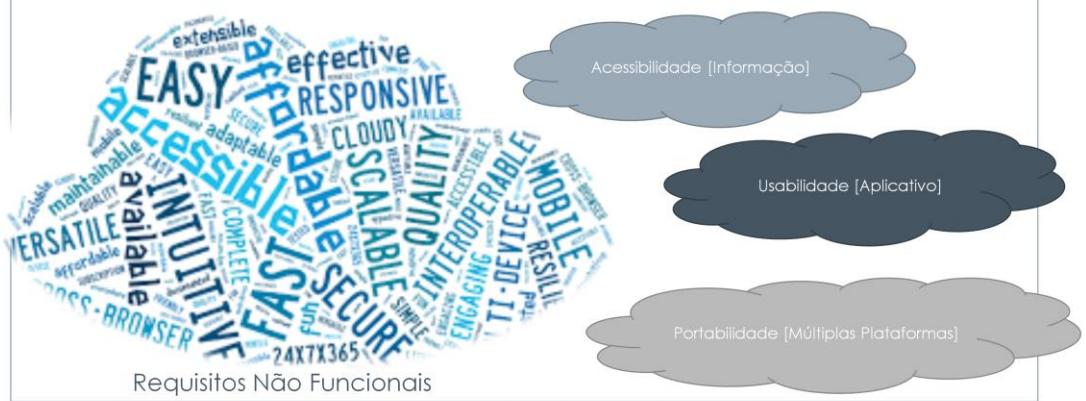
Vantagens...

Outra vantagem que chama a atenção nesse tipo de desenvolvimento é que as abstrações da orientação à meta são bem interessantes para expressar as demandas de uma organização e/ou de um sistema a ser implantado nessa organização.

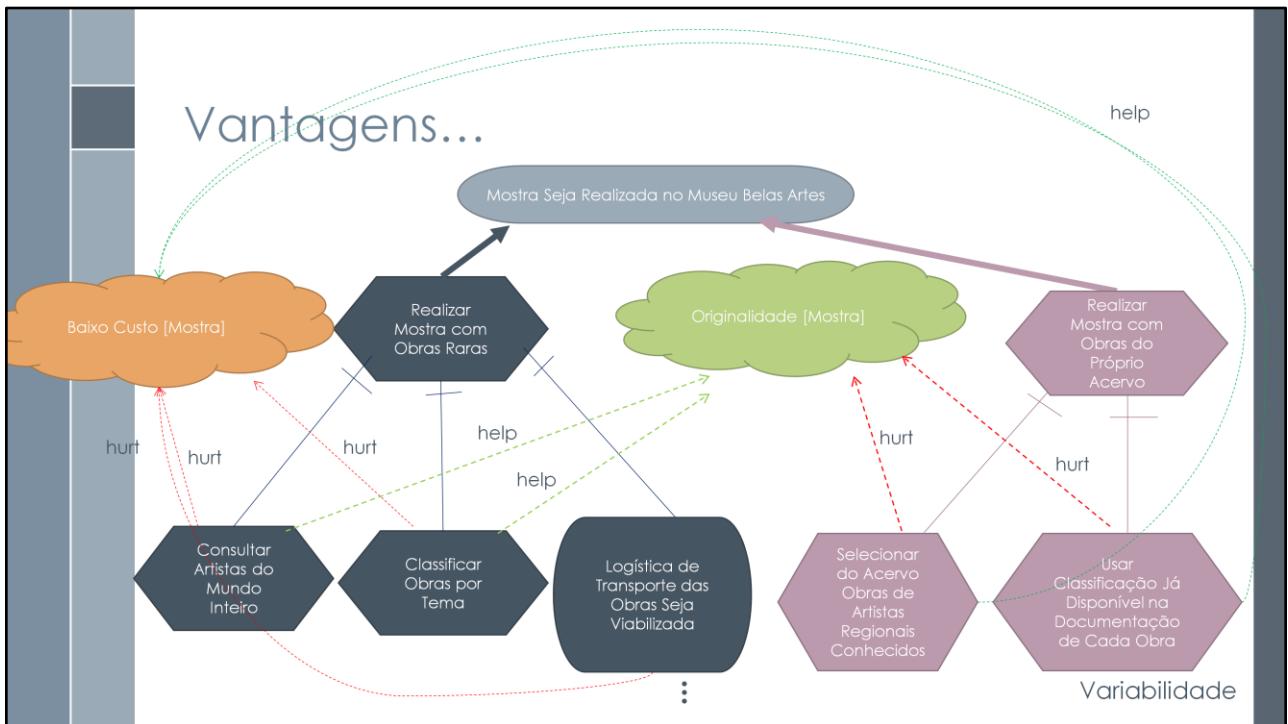


Vantagens...

Outras vantagens a serem destacadas são, principalmente: foco nos requisitos não funcionais e na variabilidade – no nível de especificação – de alternativas para se atingir uma dada meta. Vantagens essas, muito por conta dos modelos utilizados...



Vantagens...



Se bem aplicados esses recursos....

Mesmas considerações feitas para a notação i*, pois tanto TROPOS quanto i*são orientados à meta...



Desvantagens...

Dentre as desvantagens, destacam-se: não temporalidade na especificação dos modelos e complexidade da notação.
Desvantagens essas, muito por conta dos modelos utilizados...

Temporalidade Ausente



Complexidade Alta

Outras Abordagens Similares...



KAOS
MESSAGE;
Adelfe;
MASE;
Prometheus, e
Várias outras, principalmente, associadas ao paradigma de Sistemas Multiagentes.

Overview: <http://www.e-publicacoes.uerj.br/index.php/cadinf/article/download/6526/4645>

Considerações Finais

Considerações Finais

- › Nessa aula, foi apresentada a atividade de modelagem de requisitos com base em artefatos de uma abordagem mais emergente.
- › No caso, focou-se em:
 - GORE (Goal-Oriented Requirements Engineering);
 - *i**, esse como um framework conceitual, propondo uma notação, e
 - TROPOS, esse como uma metodologia orienta à meta.
- › Mais detalhes? Acessem:
 - http://istar.rwth-aachen.de/tiki-view_articles.php
 - <http://www.cs.toronto.edu/km/istar/>
 - <http://www.troposproject.eu/>
- › Continuem os estudos!



Referências

Referências

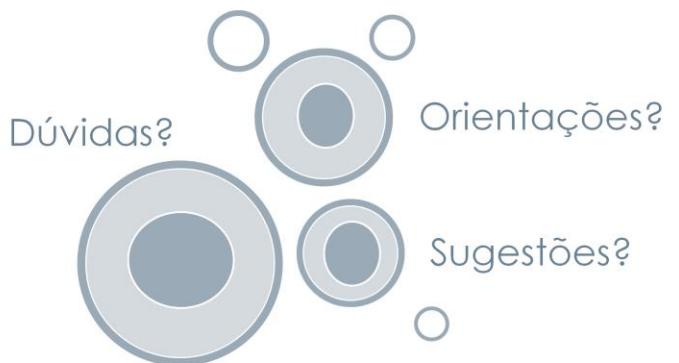
Bibliografia Básica

1. [Elibrary] Young, Ralph. Requirements Engineering Handbook. Norwood, US: Artech House Books, 2003.
2. [Open Access] Leite, Julio Cesar Sampaio do Prado. Livro Vivo - Engenharia de Requisitos. <http://livrodeengenhariaderequisitos.blogspot.com.br/> (último acesso: 2017)
3. [Elibrary] Chemuturi, Murali. Mastering Software Quality Assurance : Best Practices, Tools and Technique for Software Developers. Ft. Lauderdale, US: J. Ross Publishing Inc., 2010.
4. Software & Systems Requirements Engineering: In Practice - Brian Berenbach, Daniel Paulish, Juergen Kazmeier, Arnold Rudorfer (Livro bem completo mas, não tem exemplar físico na biblioteca, nem mesmo consta na Elibrary)
5. Requirements Engineering and Management for Software Development Projects - Murali Chemuturi (Livro bem completo mas, não tem exemplar físico na biblioteca, nem mesmo consta na Elibrary)

Referências

Bibliografia Complementar

1. [BIBLIOTECA – 15 exemplares] Pfleeger, Shari Lawrence. Engenharia de Software: Teoria e Prática. 2º. Edição. São Paulo: Prentice Hall, c2004. xix, 535 p. ISBN 978858791831
2. [BIBLIOTECA – 3 exemplares] Withall, Stephen. Software Requirement Patterns. Redmond: Microsoft Press, c2007. xvi, 366 p. ISBN 978735623989.
3. [BIBLIOTECA - vários exemplares] Leffingwell, 2011, Agile Software Requirements, <http://www.scaledagileframework.com/> (último acesso: 2017)
4. [Ebrary] Evans, Isabel. Achieving Software Quality Through Teamwork. Norwood, US: Artech House Books, 2004.
5. [Ebrary] Yu, Eric, Giorgini, Paolo, and Maiden, Neil, eds. Cooperative Information Systems: Social Modeling for Requirements Engineering. Cambridge, US: MIT Press, 2010.
6. [Open Access] Slides disponíveis <https://www.wou.edu/~eltonm/Marketing/PP%20Slides/> (último acesso: 2017) em:



FIM

mileneserrano@unb.br ou mileneserrano@gmail.com
serrano@unb.br ou serr.mau@gmail.com