3
Cenários

Cenários são considerados descrições evolutivas de situações num ambiente [9], sendo compostos por um conjunto ordenado de interações entre seus participantes. No contexto do desenvolvimento de sistemas, usualmente esta interação se dá entre o sistema em desenvolvimento e os atores externos, que podem ser usuários ou outros sistemas. Segundo Carroll [61], a propriedade que melhor define um cenário é o fato do mesmo projetar uma descrição concreta de uma atividade em que o usuário se engaja no momento em que está realizando uma tarefa específica.

Cenários têm sido amplamente utilizados no processo de elicitação dos requisitos de sistemas de software. Neste contexto, cenários são utilizados para descrever as situações de uso do sistema pelos seus usuários e os relacionamentos entre o sistema em desenvolvimento e outros sistemas externos, auxiliando no entendimento e na descoberta de novos requisitos.

A utilização de cenários em outras fases do processo de desenvolvimento de software tem sido proposta por diversos autores.

Carroll [61] acredita que cenários podem trazer benefícios em diversas outras áreas: na comunicação entre clientes e desenvolvedores, permitindo que os próprios clientes descrevam cenários que ilustrem elementos de desenho importante para eles, problemas ou novas situações que desejam que o sistema implemente; na captura das justificativas do desenho do sistema, registrando o processo de tomada de decisão através do registro das alternativas (estudadas, rejeitadas e aceitas) e dos argumentos utilizados; nas projeções futuras de uso do sistema, como meio de prototipar o funcionamento do futuro sistema, especialmente do ponto de vista da interação com clientes e usuários; no projeto (desenho) do sistema, através da avaliação de alternativas de desenho; na implementação, ilustrando a interação entre os diversos subsistemas; na documentação do sistema; no treinamento dos usuários, exemplificando o uso do

sistema; e na avaliação do sistema, verificando se os requisitos descritos nos cenários foram atendidos.

A revisão da bibliografia da utilização de cenários realizada por Filippidou [62] identificou a possibilidade de utilização de cenários na elicitação de requisitos, na projeção de requisitos, na análise dos requisitos, na avaliação dos requisitos, na captura de justificativas e na geração de interfaces.

Leite et al. [63][64] acreditam que cenários podem ser utilizados e produzidos não somente na fase de definição de requisitos, mas durante todo o processo de desenvolvimento de um sistema. Desta maneira existem desde cenários iniciais que modelam o macrosistema onde o sistema será desenvolvido e instalado até cenários que representam as interações dos usuários com o sistema quando este estiver em funcionamento, passando por várias versões intermediárias ao longo do processo de desenvolvimento do software.

Nossa visão da utilização de cenários nesta tese corresponde a este espectro mais amplo de uso, defendido por Leite et al. Neste contexto, conjuntos de cenários são criados e utilizados em todas as fases do processo de desenvolvimento. Utilizando como exemplo o ciclo de vida tradicional (cascata), seriam criados conjuntos de cenários correspondentes aos requisitos, a análise, ao projeto (desenho), a implementação, aos testes e a manutenção. Da mesma forma que os outros artefatos do processo, as informações descritas nos cenários estão em constante evolução. Mudanças realizadas em um cenário devem ser propagadas em ambas as direções. Por exemplo, mudanças nos cenários de projeto devem ser propagadas tanto para os cenários de implementação quanto para os de análise e de requisitos.

Neste capítulo será apresentada uma visão geral dos aspectos relacionados a cenários que fundamentam o sistema de rastreamento de cenários baseado em transformações a ser apresentado no capítulo 4.

3.1. Modelo de Cenário

Weidenhaupt et al. [65] realizaram um estudo sobre o uso de cenários no desenvolvimento de software e identificaram cinco formas utilizadas para descrever cenários: texto narrativo, texto estruturado, diagramas, imagens e

animações ou simulações. Dentre estas, a forma mais utilizada pelos projetos estudados foi o texto estruturado.

A notação de cenários adotada nesta tese é a proposta por Leite [9]. Esta notação utiliza uma linguagem natural semi-estruturada (ancorada no mundo real), partindo da premissa que a utilização da linguagem da aplicação e não do software facilita o entendimento e a validação dos requisitos por parte dos clientes.

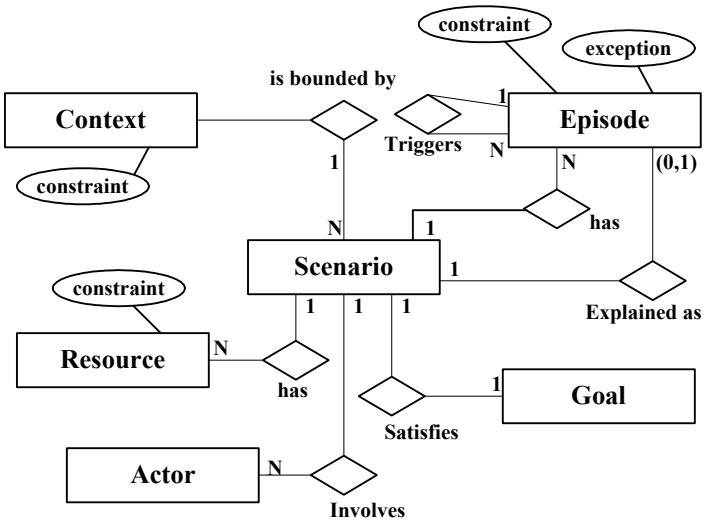

Figura 18 – Diagrama de entidades e relacionamentos para cenários

A notação para cenários utiliza a estrutura apresentada na figura 18 e composta pelos seguintes elementos:

- Título (*title*): identifica o cenário.
- Objetivo (*goal*): estabelece a finalidade de um cenário. O cenário deve descrever de que modo este objetivo deve ser alcançado.
- Contexto (*context*): descreve o estado inicial de um cenário, suas pré-condições, o local (físico) e tempo. Na sua definição podem ser especificadas restrições sobre estes elementos (*constraint*).
- Recurso (*resource*): identifica os objetos passivos com os quais lidam os atores. Na sua definição podem ser especificadas restrições sobre os objetos a serem lidados pelo cenário (*constraint*).
- Ator (*actor*): Pessoa ou estrutura organizacional que tem um papel no cenário.
- Episódio (*episode*): Cada episódio representa uma ação realizada por um ator onde participam outros atores utilizando recursos disponíveis. Um episódio também pode se referir a outro cenário. Episódios podem conter restrições (*constraint*) e exceções (*exception*). Uma restrição é

qualquer imposição que restrinja um episódio de um cenário. Uma exceção é o tratamento para uma situação excepcional ou de erro.

A figura 19 mostra um exemplo de utilização da notação para a descrição de um cenário para um sistema de controle de luzes, utilizado como estudo de caso nesta tese e descrito na seção 5.3.

Title	malfunction
Goal	how to proceed when a malfunction occurs
Context	4th floor of building 32
Actor	Control system
Resource	Ceiling light groups
Resource	Control panel
Episodes	inform facility manager of malfunction
	inform user of malfunction
	Control system supports the facility manager in finding the reasons of malfunction
	store malfunction
	facility manager can correct manually malfunction undetected by the Control system

Figura 19 – Exemplo de descrição de um cenário

Um importante complemento para a descrição dos cenários é o Léxico Ampliado da Linguagem (LAL), um metamodelo projetado para ajudar a capturar a linguagem utilizada no domínio. O LAL é apoiado em uma idéia simples: entender a linguagem do problema, sem se preocupar em entender o problema. É uma representação em linguagem natural que objetiva capturar o vocabulário da aplicação. Seu principal objetivo é registrar palavras ou frases peculiares ao domínio em questão. O LAL vai além da captura do significado (denotação) dos termos, como a maioria dos glossários, pois também inclui a conotação dos mesmos. Desta forma podemos compreender o significado dos termos de modo independente e em relação a outros termos. Cada entrada no LAL possui dois tipos de descrição. O primeiro tipo chama-se *Notion*, cujo objetivo é descrever a denotação da palavra ou frase. O segundo tipo, denominado *Behavioral Response*, objetiva descrever a conotação da palavra ou frase, isto é, informações adicionais no contexto. Na descrição destes devem ser seguidos dois princípios: Princípio da Circularidade, procurando maximizar o uso dos outros termos do LAL; e o Princípio do Vocabulário Mínimo, procurando usar o vocabulário básico de uma língua que normalmente é constituído das palavras mais comuns da língua.

Na descrição do cenário mostrado na figura 19, os termos sublinhados são os símbolos descritos no LAL. Um exemplo da definição destes símbolos no LAL

é mostrado na figura 20, onde os elementos sublinhados correspondem a outros símbolos descritos no LAL, desta maneira permitindo a visualização da aplicação do princípio da circularidade.

Name	**Malfunction**
Notion	Incorrect behavior of a device
Behavior	There can be a malfunction of the motion detector
	There can be a malfunction of the outdoor light sensor
	All malfunctions are stored and reported on request
	The facility manager has to be informed
	The control system supports the facility manager by finding the reason for a malfunction
Name	**4th floor of building 32**
Notion	Fourth floor of building 32 of the University of Kaiserslautern
	It consists of three hallway section and two staircase (SCE and SCW) to other floors of the building
Behavior	Location where a new light control system will be installed
	The architecture describes the floor description for the 4th floor of building 32
Name	**Control system**
Notion	Hardware and software control system that controls indoor climate, lighting , safety and security
	Control system active is an output of the control system
	The control system can access only digital values
Behavior	The control system should address non-functional requirements NF1 , NF4 and NF5
	The control system uses a pulse to toggle the light
	If a malfunction occurs, the control system supports the facility manager in finding the reason
	The control system provides reports on current and past energy consumption
	Malfunctions that the control system cannot detect can be entered manually
	If any outdoor light sensor does not work correctly, the control system for rooms should behave as if the outdoor light sensor had been submitting the last correct measurement of the outdoor light constantly
	If any motion detector of a room or a hallway section does not work correctly, the control system should behave as if the room or the hallway section were occupied

Figura 20 – Exemplo de definição do LAL

3.2. Evolução de Cenários

Cenários descrevem o comportamento de um sistema quando inserido no ambiente do usuário. A percepção deste comportamento evolui ao longo do processo de desenvolvimento do sistema, pois novas necessidades são identificadas, bem como aumenta o entendimento do mundo real dos usuários, tornando a evolução uma característica intrínseca dos cenários.

Este caráter evolutivo dos cenários é uma característica aceita pela maior parte dos desenvolvedores de sistemas e obriga a existência de um gerenciamento desta evolução, provendo mecanismos para manter a rastreabilidade, o controle de configuração e a eliminação de inconsistências e desatualizações entre os cenários e gerados ao longo do processo de desenvolvimento.

Breitman [10] realizou uma extensa análise sobre evolução de cenários, propondo um modelo genérico de evolução de cenários. Os componentes estáticos deste modelo referem-se aos relacionamentos existentes entre cenários de uma mesma configuração, enquanto que os componentes dinâmicos são caracterizados pelas operações sobre cenários que são aplicadas pelo desenvolvedor durante o processo de desenvolvimento do sistema, ou seja, operações aplicadas sobre os cenários de uma configuração de maneira a criar uma nova configuração. Os conjuntos de relacionamentos e operações foram organizados na taxonomia para evolução de cenários apresentada na figura 21.

Relacionamentos	**Operações**	
Aspectos Estáticos	**Aspectos Dinâmicos**	
	intra cenário	**Inter cenários**
Complemento	Inclusão	Fusão
Equivalência	Modificação	Encapsulamento
Contenção	Remoção	Consolidação
Pré-Condição		Divisão
Detour		Divisão Múltipla
Exceção		Especialização
Inclusão		
Possível Precedência		

Figura 21 – Taxonomia para evolução de cenários

Cada um dos relacionamentos entre cenários será detalhado a seguir, apresentando sua descrição e o conjunto de heurísticas utilizadas na sua identificação.

a) **Complemento:** Cenários que conjuntamente respondem a um objetivo maior.

- **Heurísticas para identificação:**
  - Cenários devem possuir objetivos idênticos.
  - Existe coincidência entre o contexto dos cenários envolvidos.
  - Existe coincidência de recursos entre os cenários. Esta coincidência pode ser total, todos os cenários apresentam o mesmo recurso ou parcial, i.e., notada entre pares dos cenários envolvidos.
  - Existe grande coincidência de atores entre os cenários envolvidos.

- Existe notadamente pequena coincidência entre os episódios de cada um dos cenários envolvidos.

b) **Equivalência:** Cenários tratam de situações semelhantes.

- **Heurísticas para identificação:**
  - Cenários devem possuir objetivos idênticos.
  - Existe coincidência entre o contexto dos cenários envolvidos.
  - Existe pequena coincidência de recursos entre os cenários
  - Existe grande coincidência de atores entre os cenários envolvidos.
  - Existe grande coincidência de episódios entre os cenário envolvidos..

c) **Contenção:** Cenários que compartilham o mesmo contexto ou o contexto de um deles é totalmente contido no contexto do segundo.

- **Heurísticas para identificação:**
  - Cenários envolvidos podem apresentar objetivos diferentes.
  - O contexto de um cenário A contido em B deve ser igual ao de B, podendo ser acrescido de algumas restrições.
  - Pode existir coincidência de recursos entre os cenários envolvidos.
  - Existe grande coincidência de atores entre os cenários envolvidos.
  - Pelo menos um episódio do cenário contenedor deve fazer parte do cenário contido.

d) **Pré-Condição:** Representa uma dimensão temporal entre cenários: para que um seja executado, outro deve ter sido executado anteriormente.

- **Heurísticas para identificação:**
  - Cenário A faz parte do contexto do cenário B
  - Existe coincidência de pelo menos um ator entre os cenários envolvidos.
  - Pode existir coincidência de episódios entre os cenários envolvidos.

e) **Detour:** Um cenário trata uma exceção de um outro cenário, retornando ao mesmo ponto do cenário original.

- **Heurísticas para identificação:**
  - Um relacionamento de Detour entre os cenários A e B se estabelece quando:
    - Cenário B é uma exceção de um dos episódios do cenário A, e
    - Cenário A é um dos episódios do cenário B.

f) **Exceção:** Um cenário trata uma exceção de um cenário.

- **Heurísticas para identificação:**

- Existe um relacionamento de exceção entre os cenários A e B quando:
- Existe uma exceção em um dos episódios do cenário A, e esta exceção é o próprio cenário B.

g) **Inclusão:** Um cenário utiliza um outro cenário como um dos seus passos.

- **Heurísticas para identificação:**

- Pelo menos um episódio é compartilhado por todos os cenários envolvidos.

h) **Possível Precedência:** Quando a partir de uma situação de exceção de um cenário inicial é necessário fixar a ordem em que os cenários resultantes devem seguir.

- **Heurísticas para identificação:**

- Pode existir coincidência entre os objetivos dos cenários envolvidos.
- Existe coincidência entre o contexto dos cenários envolvidos.
- Existe coincidência de pelo menos um dos atores.

As operações sobre cenários podem ser classificadas como intra cenário - aquelas que se referem a modificações realizadas em um único cenário - ou inter cenários - que agem sobre um conjunto de cenários. A figura 22 detalha as operações que podem ser aplicadas sobre os cenários.

Operações Inter-Cenários	
Fusão	Objetiva a união de procedimentos em um único cenário
Encapsulamento	União de 2 ou mais cenários que apresentam muita coincidência de conteúdo
Consolidação	Objetiva aumentar o escopo de um único cenário através de generalização
Divisão	Divide o conteúdo de um cenário em dois ou mais cenários independentes
Múltipla Divisão	Isolar comportamento comum partilhado por diversos cenários em um único cenário
Especialização	Incorporar cursos alternativos para um único cenário genérico
Extensão	Caso especial da especialização
Exclusão	Retirar informações obsoletas
Adição	Introduzir um novo cenário
Operações Intra-Cenário	
Inclusão	Incluir novas informações no cenário
Modificação	Modificar informações no cenário
Retirada	Retirar informações do cenário

Figura 22 – Operações sobre cenários

3.3. Rastreamento de Cenários

Um modelo de rastreamento deve representar o processo de modificação, refinamento e evolução dos diferentes artefatos oriundos do desenvolvimento de software. O objetivo do modelo é permitir a captura da história de

desenvolvimento de uma maneira estruturada de forma a facilitar o seu entendimento e permitir seu gerenciamento.

Um sistema de rastreamento pode ser definido como uma rede semântica na qual seus nós representam objetos e sobre os quais é estabelecido o rastreamento através de ligações de diversos tipos e intensidade. Um sistema de rastreamento deve, através de seus nós e relacionamentos, permitir a captura das diversas dimensões de informações [3]:

- Qual informação deve ser armazenada, definindo os atributos e as características necessárias das informações;
- Quem são as pessoas (*stakeholders*) que executam os diferentes papéis na criação, manutenção e uso dos vários objetos e de seus relacionamentos;
- Onde a informação deve ser representada definindo os artefatos que irão formalizar as informações de rastreamento;
- Como a informação deve ser representada através de mecanismos formais e informais e como se relaciona com os outros componentes do sistema;
- Porque algum objeto foi criado, modificado ou evoluído; e
- Quando a informação foi capturada, modificada e evoluída.

Segundo Ramesh & Jarke [3], existem duas categorias de usuários de um sistema de rastreamento quanto ao objetivo almejado com seu uso: low-end users que engloba os usuários que possuem pouca experiência no uso de rastreamento e o utilizam para se adequar a algum padrão de desenvolvimento ou como uma exigência gerencial; e high-end users que são usuários experientes que utilizam o sistema objetivando aumentar a satisfação do usuário e criar uma base de conhecimento ao longo do ciclo de vida do sistema. A migração dos usuários da primeira categoria para a segunda se dá não apenas devido ao aumento da experiência dos usuários, mas principalmente devido ao crescimento da complexidade dos sistemas em desenvolvimento.

3.3.1. Classificação das Ligações de Rastreamento

Existem diversas propostas para classificar as ligações de rastreamento entre os diversos tipos de artefatos oriundos do processo de desenvolvimento de um sistema.

3.3.1.1. Modelo de Referência de Ramesh & Jarke

Ramesh & Jarke [3] realizaram um estudo que analisou 26 grandes organizações de desenvolvimento de software quanto aos métodos e ferramentas de rastreamento de requisitos utilizados, apresentando um modelo de referência para rastreamento de requisitos. A figura 23 mostra este modelo de referência, que classifica as ligações entre os objetos como: Ligação Satisfaz (Satisfies Link), que especifica que um objeto de alto nível define alguma restrição, política ou objetivo a ser satisfeito por um objeto de um nível mais baixo; Ligação Depende (Depends Link), que especifica a dependência que um objeto tem com outro objeto de maneira a cumprir determinada restrição ou objetivo; Ligação Evolui-Para (Evolves-To Link); que especifica que um objeto evolui para outro objeto através de alguma ação executada; e a Ligação Justifica (Rationale Link), que especifica a razão pela qual se deu determinada evolução de um objeto em outro. Os dois primeiros tipos de ligações são relacionados ao produto e descrevem as propriedades e relacionamentos de objetos de design independentemente de como foram criados, enquanto as duas últimas são relacionadas ao processo, só podendo ser capturados a partir da observação da história das ações realizadas no processo.

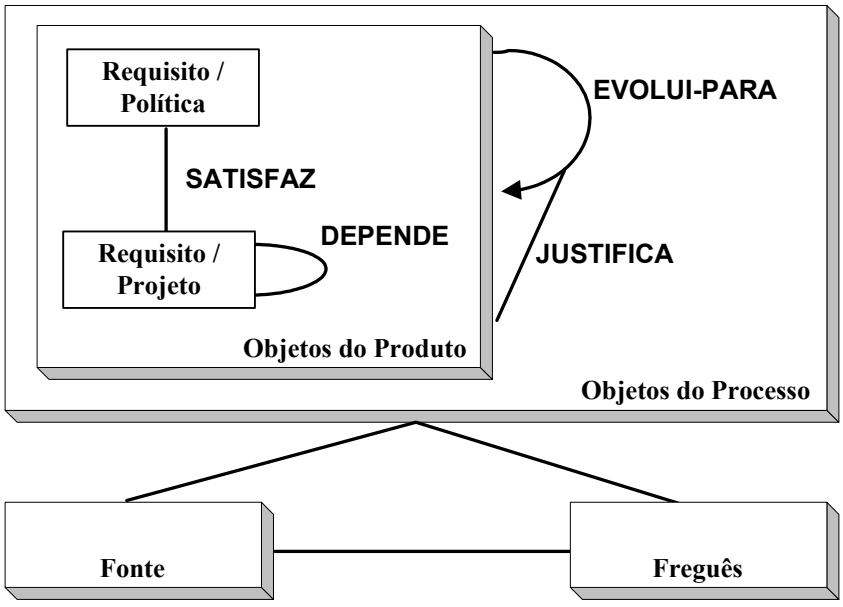

Figura 23 – Modelo de referência segundo Ramesh e Jarke

Será apresentado a seguir um detalhamento de cada um dos tipos de ligações, abordando seus objetivos e principais formas de utilização.

Ligação Satisfaz

O objetivo principal da utilização de um sistema de rastreamento é assegurar que os requisitos são satisfeitos pelo sistema, ou seja, que todas as funcionalidades necessárias a concretização dos requisitos foram implementadas pelo conjunto dos artefatos criados no processo. Este objetivo é alcançado através da utilização de ligações do tipo Satisfaz que podem ser explicitamente definidas ou serem derivadas indiretamente. De maneira a alcançar este objetivo principal, estas ligações devem ainda atender as seguintes finalidades:

- assegurar a consistência entre os resultados das diferentes fases do processo de desenvolvimento;
- rastrear os modelos projetados para satisfazer os requisitos;
- rastrear os componentes, subsistemas ou sistemas que tiveram um subconjunto dos requisitos alocados a eles;
- identificar os componentes, subsistemas ou sistemas que satisfazem os requisitos, bem como todos os seus componentes que igualmente os satisfazem;
- rastrear as rotinas de verificação dos requisitos, tais como testes ou simulações, especificadas para cada requisito; rastrear o resultado da aplicação das rotinas de verificação de maneira a assegurar que os requisitos foram realmente satisfeitos.

Ligação Evolui-Para

As ligações Evolui-Para são responsáveis pela documentação dos relacionamentos entre as entradas e saídas das ações realizadas entre um artefato existente e um novo artefato ou o artefato modificado. Seu uso principal é na identificação da origem dos vários objetos de maneira a permitir um melhor entendimento dos requisitos e rastrear o histórico de modificações e refinamentos dos artefatos.

Ligação Justifica

As ligações Justifica são utilizadas na representação dos motivos que levarão a criação e modificação dos artefatos do desenvolvimento, ou seja, as justificativas para os passos evolucionários realizados no desenvolvimento do sistema. Estas ligações são utilizadas para que se identifique claramente:

- as justificativas para a criação e modificação dos artefatos;

- as decisões realizadas bem como as alternativas analisadas e aquelas que foram descartadas, permitindo um perfeito entendimento da solução adotada e desta maneira facilitando o reuso e manutenção do sistema;
- as hipóteses consideradas no desenvolvimento; e
- o contexto no qual os artefatos foram criados.

Ligação Depende

O objetivo das ligações Depende é auxiliar na gerência das dependências entre os artefatos, normalmente dentro de uma mesma etapa do desenvolvimento. Seu uso se dá rastreando a composição e a hierarquia de artefatos, bem como na análise da propagação das mudanças realizadas em um artefato.

3.3.1.2. Classificação quanto ao conteúdo proposta por Hughes & Martin

A figura 24 mostra a classificação proposta por Hughes e Martin[66] para as ligações de Rastreamento, que utiliza o critério do conteúdo para classificar as ligações em Rastreamento de Requisitos e Rastreamento de Projeto.

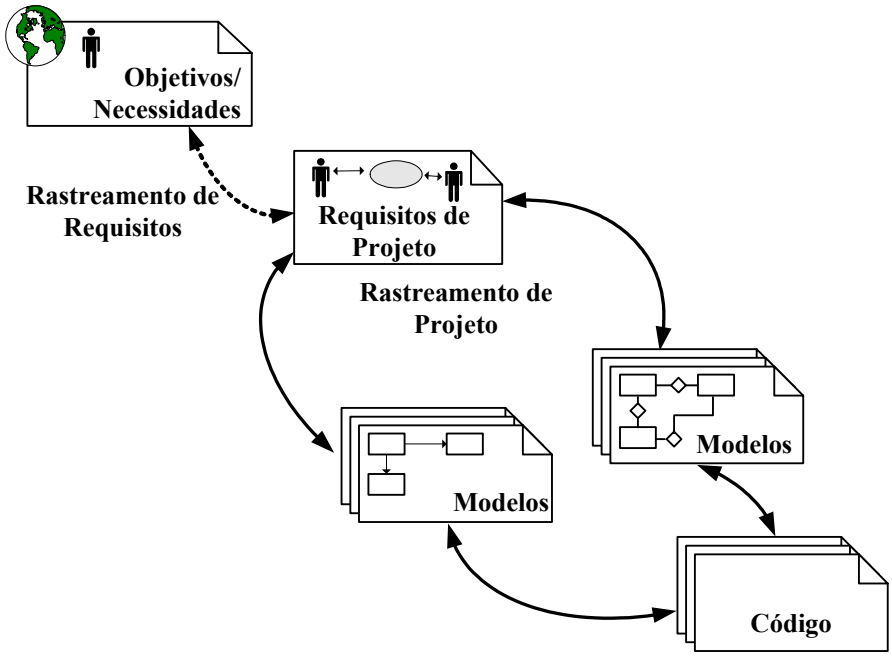

Figura 24 - Classificação das ligações de rastreamento quanto ao conteúdo

As ligações de Rastreamento de Requisitos relacionam os requisitos definidos em termos de objetivos e necessidades dos usuários com os requisitos de projeto do sistema em desenvolvimento. Os requisitos de projeto são derivados dos objetivos e necessidades do usuário através da análise e decomposição do domínio do problema.

As ligações de Rastreamento de Projeto relacionam os requisitos de projeto com as suas manifestações físicas. Desta maneira elas representam o processo de tomada de decisão realizado nesta transformação.

3.3.1.3. Classificação quanto ao sentido proposta por Gotel e Finkelstein

Gotel & Finkelstein[1] classificam as ligações de rastreamento de acordo com seu sentido em relação aos requisitos do sistema, classificando-os em:

- Pré-Rastreamento de Requisitos, que relaciona os requisitos a sua origem, ou seja, aos documentos do mundo real que serviram de base na sua definição; e
- Pós-Rastreamento de Requisitos, que mostra a realização dos requisitos, relacionando-os aos modelos e código fonte que os implementam.

A figura 25 mostra estes tipos de ligação. Cada um deles pode ainda ser dividido de acordo com o sentido em retroceder de/para os requisitos (*backward from/to requirements*) ou avançar de/para os requisitos (*forward to/from requirements*).

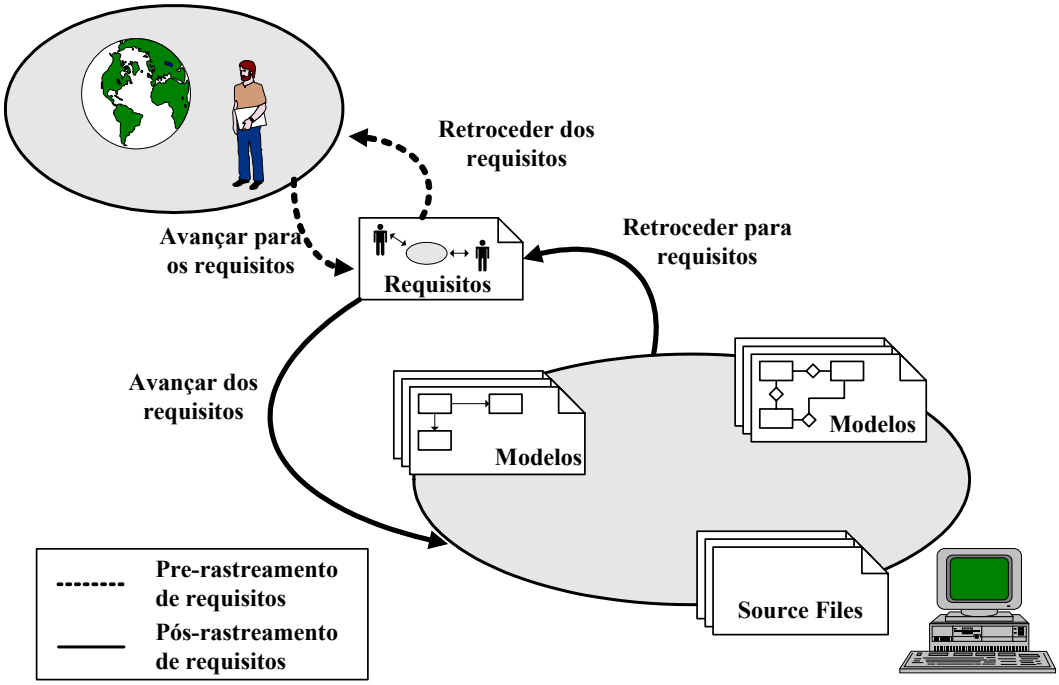

Figura 25 - Classificação das ligações de rastreamento quanto ao sentido

Haumer et al. [40] apresentam um refinamento da proposta de Gotel e Finkelstein definindo um novo tipo de ligação de Pré-Rastreamento de Requisitos chamado de Pré-Rastreamento de Requisitos Estendido. A figura 26 mostra este tipo de ligação que é criado para relacionar os requisitos a exemplos concretos do

mundo real utilizando um detalhamento maior na sua representação, permitindo desta maneira o estabelecimento de inter-relacionamentos entre partes dos requisitos e porções arbitrárias de observações do mundo real.

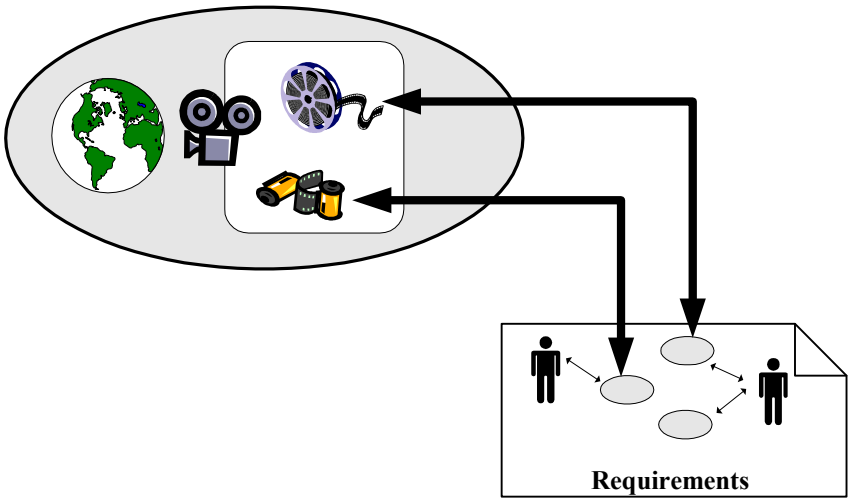

Figura 26 - Pré-Rastreamento de Requisitos Estendido

3.3.2. Requisitos de um sistema de rastreamento

Ramesh & Jarke apresentam como conclusão de seu estudo um conjunto de características que os sistemas de rastreamento devem possuir para permitir seu uso efetivo pelas diversas categorias de usuários, bem como uma visão geral da situação geral das ferramentas existentes em relação a estes requisitos.

Representação da Ligações

Um sistema de rastreamento deve permitir a representação da semântica das ligações e dos atributos necessários a obtenção de um maior nível de suporte automático a manipulação dos diversos tipos de ligação. Dentre estes atributos destacam-se a intensidade da ligação, sua criticalidade e sua importância. Por exemplo, no caso das ligações do tipo Satisfaz é necessário representar o quanto um determinado artefato está sendo satisfeito por algum outro. Já no caso das ligações do tipo Evolui-Para devem poder ser representadas a medida de sua importância, de maneira a que se possa variar a granularidade da justificativa relacionada, bem como as diversas alternativas existentes para evoluir um artefato, e não apenas armazenar a alternativa utilizada. Para as ligações Depende é necessário representar o sentido e a intensidade da dependência, ou seja, o quanto um artefato depende do outro.

Em relação ao suporte existente atualmente para representar a semântica das ligações, a maior parte dos sistemas de rastreamento armazena apenas a informação que um artefato evoluiu para outro, sem qualquer elaboração. Outras ferramentas tratam somente a semântica da ligação, ou seja, tratam todas as ligações de um mesmo tipo como igualmente críticas.

Variação da Granularidade

Um sistema de rastreamento deve possuir mecanismos de abstração de tal maneira que seja possível variar a granularidade e a sofisticação das tarefas de rastreamento. Por exemplo, no caso das Ligações de Justificativa, utilizadas para descrever as razões que levaram a modificação de um artefato, a granularidade deve poder variar de acordo com o tipo dos artefatos relacionados bem como com a intensidade ou criticidade do passo evolutivo. Uma outra característica desejável é possibilitar que sejam estabelecidas ligações entre os artefatos e as fontes de informação, de maneira a aumentar a granularidade necessária, diminuir o trabalho necessário para descrever a justificativa e evitar a perda de detalhes.

A maior parte das propostas existentes permite a utilização apenas de um nível de granularidade fixo, algumas armazenando apenas o tipo da ligação (granularidade grossa), enquanto que outras utilizam um detalhamento muito grande (granularidade fina).

Serviços de Inferência

Disponibilizar serviços de inferência com suporte a semântica dos diversos tipos de ligações que permitam: a navegação pela base de dados, a obtenção de informações relevantes, a manutenção da integridade da base através da propagação de informações e a dedução de novas informações. Por exemplo, no caso de ligações Depende e Satisfaz se um artefato depender de um segundo, e o segundo depender de um terceiro, a dependência (ou satisfação) do primeiro artefato com o terceiro deve ser automaticamente inferida. Já para as ligações do tipo Evolui-Para, o sistema de rastreamento deve apresentar a solução ótima ou fazer inferências à cerca de "se eu implementar desta forma, o quanto estou satisfazendo meus requisitos".

Muitas ferramentas não oferecem serviços de inferência, principalmente por não armazenarem a semântica da ligação, requisito fundamental para a disponibilização do serviço.

Flexibilidade de Representação

Permitir o uso tanto de representações formais quanto de representações informais das informações de rastreamento.

Gerência de Configuração

Os sistemas de rastreamento devem permitir a definição de versões e o gerenciamento da configuração do sistema, bem como possuir um mecanismo de notificação de mudanças.

Integração com o Processo de Desenvolvimento

Integrar o rastreamento ao processo de desenvolvimento de maneira a definir qual a informação a ser capturada, como a informação deve ser armazenada e o inter-relacionamento entre as atividades do processo de desenvolvimento e os passos de rastreamento.