5
Uso do Mecanismo de Rastreamento Baseado em Transformações na Evolução de Cenários

Um dos maiores desafios no desenvolvimento e evolução de sistemas deriva do fato de que os requisitos são constantemente modificados ou evoluídos durante o ciclo de vida do sistema, fruto principalmente da necessidade de correção de uma deficiência do sistema ou de mudanças das necessidades dos seus usuários. Este caráter volátil dos requisitos obriga que o sistema de rastreamento de requisitos seja capaz de auxiliar na documentação, entendimento e gerenciamento da evolução dos requisitos, necessidade esta potencializada em sistemas que possuem um ciclo de vida longo.

O mecanismo proposto no Capítulo 4 pode ser utilizado na construção de sistemas de rastreamento para diversos tipos de artefatos oriundos do processo de desenvolvimento de software. Neste capítulo será apresentada a utilização do mecanismo na construção de um sistema de rastreamento para o processo de desenvolvimento de software baseado em cenários, apresentado no capítulo 3. Este processo foi escolhido devido à existência de uma taxonomia bem definida para evolução de cenários, proposta por Breitman [73], e a disponibilidade de informações pré-existentes que poderiam ser usadas na validação das informações capturadas pelo sistema.

A principal característica deste sistema é a captura semi-automática das informações de rastreamento através da análise de duas configurações consecutivas de cenários, da identificação de suas diferenças e do reconhecimento semi-automático das operações realizadas sobre os cenários da configuração inicial e que levaram a nova configuração. O conjunto de operações reconhecidas é o apresentado na seção 3.2. A seção 5.3 apresenta os resultados da utilização deste sistema em um estudo de caso. Concluindo o capítulo, a seção 5.4 apresenta nossas conclusões parciais sobre a viabilidade da aplicação do mecanismo proposto na captura das informações sobre rastreamento de cenários.

5.1.
Descrição do Sistema

O sistema de rastreamento para o desenvolvimento de software baseado em cenários utiliza o mecanismo apresentado no capítulo 4 na identificação semi-automática da operação realizada pelo desenvolvedor sobre os cenários. Cada operação identificada agrega um conjunto de transformadores equivalentes às ações realizadas pelo desenvolvedor na aplicação da operação. Este transformador agrega todas as informações de rastreamento de uma ligação de evolução, capturando não o conjunto de atributos que a descrevem, mas o próprio conjunto de ações executado pelo desenvolvedor sobre os cenários. A configuração original dos cenários e as operações identificadas são armazenadas e gerenciadas por um Gerenciador de Configuração que pode, a qualquer momento, recriar qualquer configuração através da aplicação das transformações necessárias.

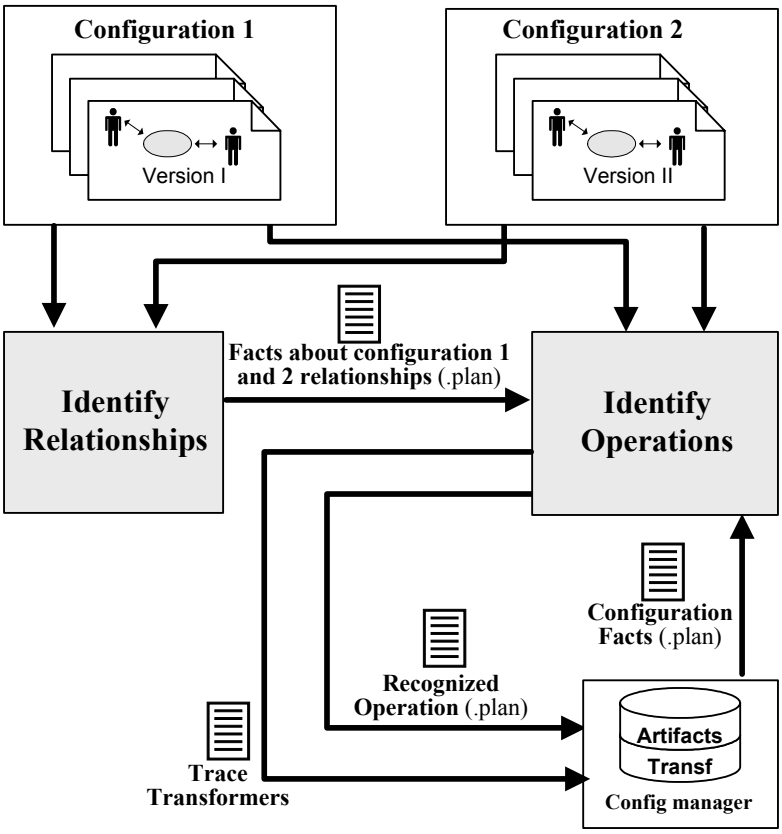

Figura 61 – Visão geral do sistema de rastreamento para o desenvolvimento de software baseado em cenários

A identificação das operações sobre cenários depende, além das informações sobre as diferenças existentes entre os cenários, de informações sobre os relacionamentos existentes entre eles. A figura 61 apresenta uma visão geral da

solução proposta. A atividade de Identificação de Relacionamentos (*Identify Relationship*) é responsável pela identificação dos relacionamentos entre os cenários de uma mesma configuração e pela geração dos fatos observados correspondentes. A atividade de Identificação das Operações (*Identify Operations*) utiliza os fatos observados sobre os relacionamentos e sobre a configuração do sistema para capturar as informações de rastreamento sobre a evolução dos cenários e armazená-las em um gerente de configuração. Estas informações de rastreamento são compostas pelas operações reconhecidas e pelos transformadores de rastreamento gerados.

Nas próximas seções serão detalhadas as atividades de Identificação de Relacionamentos e de Identificação das Operações.

5.1.1. Identificação de Relacionamentos

O objetivo desta atividade é encontrar os relacionamentos existentes entre os cenários de uma mesma configuração. Estes relacionamentos são identificados a partir de heurísticas sobre similaridades existentes entre os diversos elementos que compõem os cenários (objetivos, recursos, contextos, atores e episódios). Por exemplo, um relacionamento de Complemento entre dois cenários pode ser identificado se estes cenários tiverem:

- Alta similaridade de objetivos;
- Alta similaridade de atores; e
- Pequena similaridade de episódios.

A figura 68 apresenta o conjunto completo de relacionamentos e as heurísticas utilizadas na sua identificação.

O processo utilizado para identificar os relacionamentos existentes é mostrado na figura 62. O primeiro passo deste processo é a procura pelas similaridades (*Find Similarities*) existentes entre os cenários de uma configuração. O resultado deste passo é descrito na forma de fatos observados sobre similaridades entre os nós que representam elementos de um mesmo tipo (objetivos, recursos, atores, contextos e episódios) dos cenários em questão. Cada fato observado especifica qual o elemento do cenário e qual seu nível de similaridade. O segundo passo do processo utiliza a técnica de reconhecimento de planos para, a partir da biblioteca de planos que captura as heurísticas sobre

relacionamentos, inferir quais os relacionamentos que podem explicar o conjunto dos fatos observados.

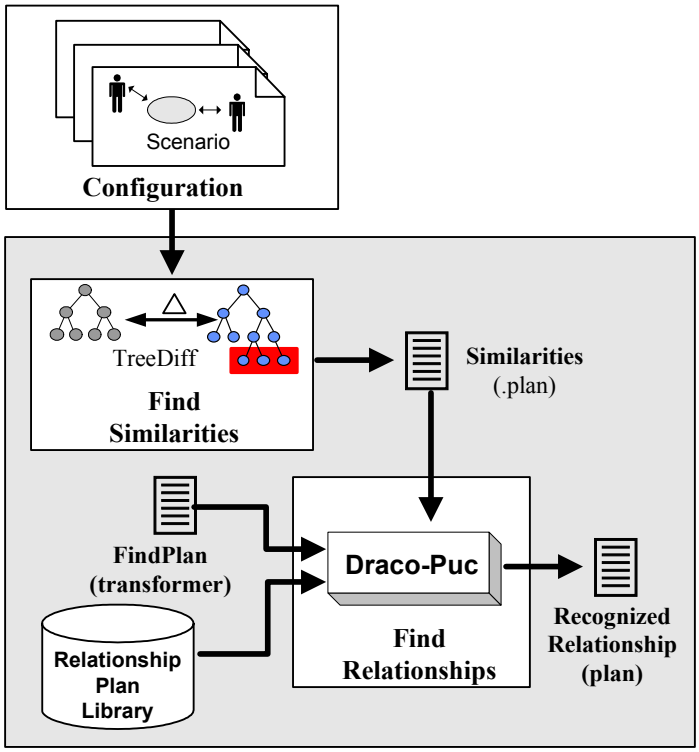

Figura 62 – Processo de identificação dos relacionamentos entre cenários

Para encontrar o nível de similaridade de cada elemento é aplicado inicialmente o algoritmo *TreeDiff* sobre cada par de cenários, sendo obtida uma lista das diferenças existentes entre estes cenários. A similaridade é inferida considerando que, se os nós não são completamente diferentes, então eles possuem algum nível de similaridade. Desta maneira podemos dizer que existe alguma similaridade entre dois nós A e B quando:

- o nó A pertence ao primeiro cenário e corresponde ao nó B do segundo cenário;
- os nós A e B representam o mesmo tipo de elemento do cenário;
- o nó A não foi removido do primeiro cenário, ou seja, não existe uma diferença relativa a remoção deste nó do primeiro cenário; e
- o nó B não foi inserido no segundo cenário, ou seja, não existe uma diferença relativa a sua inserção no segundo cenário.

O nível de similaridade entre os nós é calculado de maneira recursiva através das fórmulas apresentadas na figura 63. Se o nó tiver algum filho, este nível é a média dos níveis de similaridade dos seus filhos (equação 1), caso contrário é o seu nível atômico de similaridade (equação 2). O nível atômico de

similaridade corresponde ao nível das folhas da árvore e será igual a zero, se este nó foi inserido ou removido (equação 3), igual a uma constante se for uma mudança de referência por outra (equação 4) e, finalmente, se for uma mudança de texto, será igual à distância de edição (ED) entre o texto original e o novo texto (equação 5). A distância de edição do texto corresponde ao menor número de operações de inserção, substituição ou remoção de caracteres necessários para transformar um texto no outro, tendo sido utilizado para seu cálculo o algoritmo proposto por Marzal e Vidal [74].

① $S(E) = \frac{\sum_{i=1}^{n} S(f_i)}{n}$ *para* $n > 0$	**Onde:** n **= número de filhos de E** f_i **= filho número i**
② $S(E) = S_A(E)$ *para* $n = 0$	
③ $S_A(E) = 0$ *se* *inserção* $\wedge$ *remoção*	
④ $S_A(E) = k_1$ *se* *referência*	
⑤ $S_A(E) = ED(E)$ *se* *texto*	

Figura 63 – Equações para o cálculo do nível de similaridade de um nó

Para a definição do valor referente à mudança de uma referência por outra (k_1 na equação 4 da figura 63) é levada em consideração à noção dos termos do LAL que estão sendo referenciados: se um dos termos for a generalização do outro, for uma parte (decomposição) do outro ou se não estão relacionados. Estes valores são configuráveis e os valores utilizados no estudo de caso da seção 5.3 foram, respectivamente, 0,9 , 0.9 e 0,3.

A figura 64 apresenta um exemplo que será utilizado para exemplificar o cálculo do nível de similaridade. Os nós marcados são aqueles que possuem alguma diferença entre a árvore original (figura 64-A) e a nova árvore (figura 64-B) de um cenário: mudança de uma referência por outra (nó 3); mudança do texto (nó 6); e inclusão de um novo nó (nó 7 do novo cenário). O cálculo do nível de similaridade para as folhas das árvores são os seguintes:

- troca de uma referência por outra no nó 3. Como os elementos referenciados (X e Y) possuem um relacionamento de ISA, conforme o LAL mostrado na figura 64-C, o nível de similaridade é calculado através da equação 4 da figura 63 e corresponde a 0,9;

- mudança do texto no nó 6. Neste caso é calculada a distância de edição entre os dois textos e o resultado corresponde ao seu nível de similaridade (equação 5 da figura 63);
- inserção do nó 7. Neste caso o nível de similaridade é 0 (equação 3 da figura 63);
- o nó 4 se manteve inalterado, portanto seu nível de similaridade é 1.

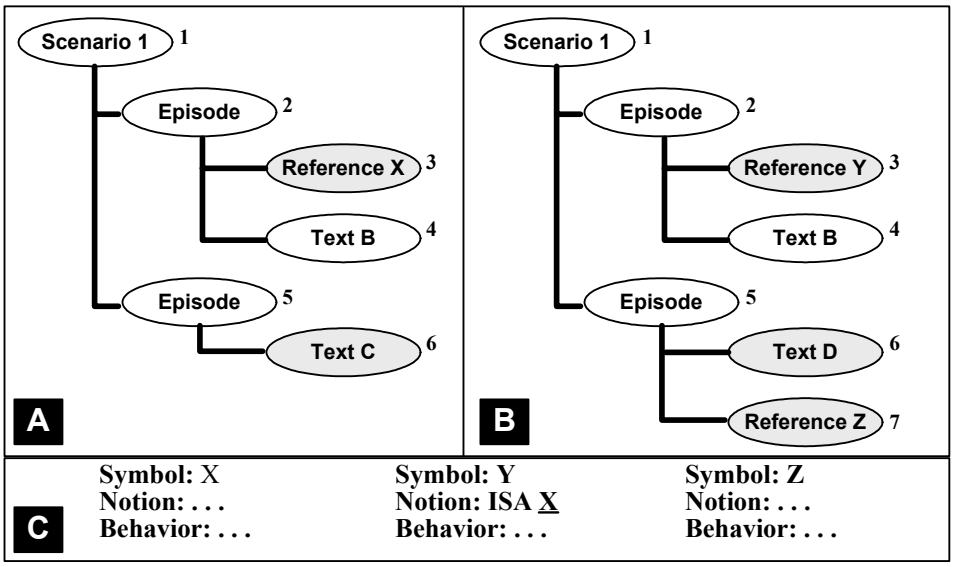

Figura 64 – Exemplo de cálculo do nível de similaridade. A) Árvore do cenário original. B) Árvore do novo cenário. C) LAL dos termos utilizados

O nível de similaridade para os nós que não são folhas (nós 2 e 5) corresponde a média dos níveis de seus filhos. Finalmente, o nível de similaridade para cada tipo de elemento dos cenários (objetivos, recursos, atores, contextos e episódios) corresponde à média dos nós do mesmo tipo. No exemplo da figura 64, o nível de similaridade para os episódios corresponde a média dos valores obtidos para os nós 2 e 5.

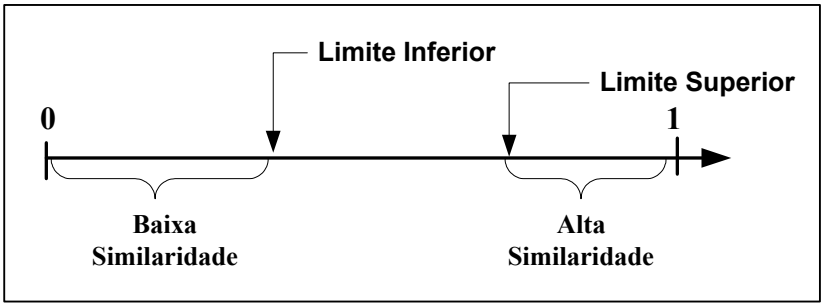

Figura 65 – Níveis de similaridade para a geração dos fatos observados

Os fatos observados sobre similaridades existentes entre os elementos dos cenários podem ser de alta similaridade ou de baixa similaridade. Uma baixa similaridade corresponde a um nível entre 0 e um limite inferior pré-estabelecido,

enquanto que uma alta similaridade corresponde a um nível calculado maior que um limite superior pré-estabelecido, conforme mostrado na figura 65. A figura 66 apresenta a parte da biblioteca de planos que especifica os diversos fatos que podem ser observados sobre as similaridades existentes entre os cenários.

```
PlanLibrary
begin
  Event subSetContextSimilarity(A,B) ;
  Event high_contextSimilarity(A,B)
  begin
    isa subSetContextSimilarity
  end
  Event low_contextSimilarity(A,B) ;
  Event contextInsideContext(A,B)
  begin
    isa subSetContextSimilarity
  end
  Event scenarioInsideContext(A,B) ;
  Event high_goalSimilarity(A,B) ;
  Event low_goalSimilarity(A,B) ;
  Event high_actorSimilarity(A,B)
  begin
    isa oneOrMoreActorSimilarity
  end
  Event oneOrMoreActorSimilarity(A,B) ;
  Event high_resourceSimilarity(A,B) ;
  Event low_resourceSimilarity(A,B) ;
  Event high_episodeSimilarity(A,B) ;
  Event low_episodeSimilarity(A,B) ;
  Event scenarioIsExceptionOfScenario(A,B) ;
  Event ScenarioIsEpisodeOfScenario(A,B) ;
end
```

Figura 66 – Biblioteca de Planos de Relacionamentos – Parte I: fatos observáveis

Serão utilizados os cenários *malfunction occurs* – Versão I (figura 67) e *malfunction* – Versão I (figura 68) para exemplificar esta atividade. A figura 69 mostra o resultado da aplicação do algoritmo *TreeDiff* sobre estes cenários, onde podemos identificar a remoção de um ator (*Facility Manager*) e mudanças nas definições de dois episódios.

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 67 – Cenário *malfunction* – Versão I

Title	malfunction occurs
Goal	how to proceed when a malfunction occurs
Context	4th floor of building 32
Actor	Control system
Actor	facility manager
Resource	Ceiling light groups
Resource	Control panel
Episodes	inform facility manager of malfunction inform user of malfunction IF Hallway section is controllable neither automatically nor manually THEN Ceiling light groups have to be on Find Reason for malfunction facility manager can correct manually malfunction s undetected by the Control system

Figura 68 – Cenário *malfunction occurs* – Versão I

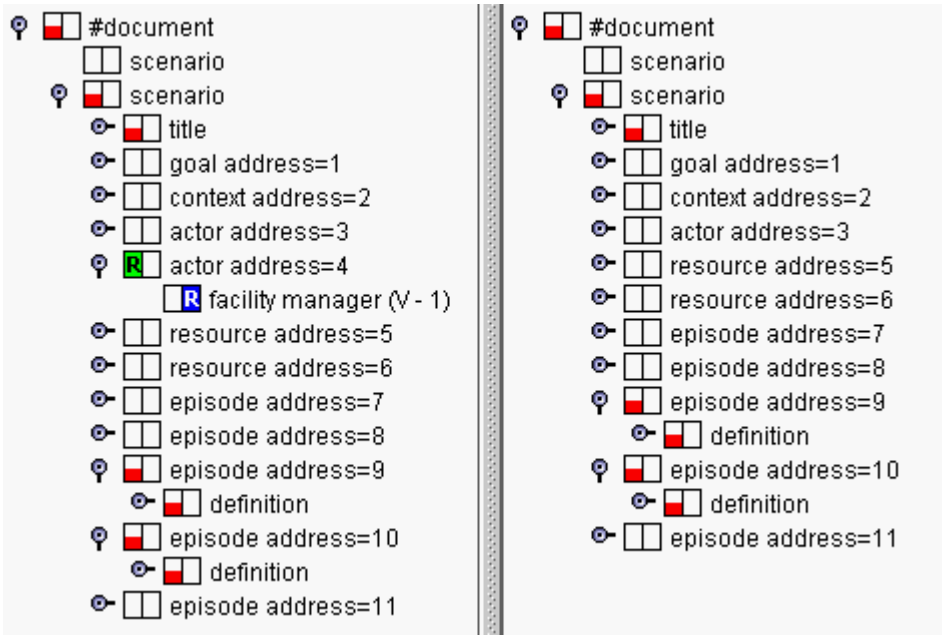

Figura 69 – Diferenças encontradas entre os cenários *malfunction occurs* – Versão I (esquerda) e *malfunction* – Versão I (direita)

A partir das diferenças encontradas entre os cenários são gerados os seguintes fatos observados sobre as similaridades existentes entre os dois cenários:

high_goalSimilarity(‘malfunction occurs’:1 , ‘malfunction’:1) ['1.0']
high_contextSimilarity(‘malfunction occurs’:1 , ‘malfunction’:1) ['1.0']
high_episodeSimilarity(‘malfunction occurs’:1 , ‘malfunction’:1) ['0.3099999']
high_resourceSimilarity(‘malfunction occurs’:1 , ‘malfunction’:1) ['1.0']

O próximo passo a ser executado é a aplicação da técnica de reconhecimento de planos para identificar qual plano existente na biblioteca de planos pode ser utilizado para explicar as similaridades encontradas entre os cenários.

Tipo	Heurística	Evento associado
Complemento	Cenários devem possuir objetivos idênticos.	*high_goalSimilarity* (peso=2,0)
	Existe coincidência entre o contexto dos cenários envolvidos.	*high_contextSimilarity* (peso=1,5)
	Existe coincidência parcial ou total de recursos entre os cenários.	*high_resourceSimilarity* (peso=1,0)
	Existe grande coincidência de atores entre os cenários envolvidos.	*high_actorSimilarity* (peso=1,5)
	Existe pequena coincidência entre os episódios de cada um dos cenário envolvidos.	*low_episodeSimilarity* (peso=4,0)
Equivalência	Cenários devem possuir objetivos idênticos.	*high_goalSimilarity* (peso=2,0)
	Existe coincidência entre o contexto dos cenários envolvidos.	*high_contextSimilarity* (peso=1,5)
	Existe pequena coincidência de recursos entre os cenários	*high_resourceSimilarity* (peso=1,0)
	Existe grande coincidência de atores entre os cenários envolvidos.	*high_actorSimilarity* (peso=1,5)
	Existe grande coincidência de episódios entre os cenários envolvidos..	*high_episodeSimilarity* (peso=4,0)
Contenção	Cenários envolvidos podem apresentar objetivos diferentes.	*low_goalSimilarity* (peso=1,5)
	O contexto de um cenário A contido em B deve ser igual ao de B, podendo ser acrescido de algumas restrições.	*subSetContextSimilarity* (peso=3,0)
	Existe grande coincidência de atores entre os cenários envolvidos.	*high_actorSimilarity* (peso=2,5)
	Pelo menos um episódio do cenário contendor deve fazer parte do cenário contido.	*low_episodeSimilarity* (peso=3,0)
Pré-Condição	Cenário A faz parte do contexto do cenário B	*scenarioInsideContext* (peso = 4
	Existe coincidência de pelo menos um ator entre os cenários envolvidos.	*high_actorSimilarity* (peso = 3)
	Pode existir coincidência de episódios entre os cenários envolvidos.	*low_episodeSimilarity* (peso = 3)
Detour	Existe um relacionamento entre dois cenários A e B quando o cenário B é uma exceção de um dos episódios do cenário A e o cenário A é um dos episódios do cenário B	*ScenarioIsEpisodeOfScenario* (peso = 1) *scenarioIsExceptionOfScenario* (peso=1)
Exceção	Existe um relacionamento entre dois cenários A e B quando existe uma exceção em um dos episódios do cenário A, e esta exceção é o próprio cenário B	*scenarioIsExceptionOfScenario* (peso=10)
Inclusão	Pelo menos um episódio é compartilhado por todos os cenários envolvidos.	*ScenarioIsEpisodeOfScenario* (peso=10)
Possível Precedência	Pode existir coincidência entre os objetivos dos cenários envolvidos.	*low_goalSimilarity* (peso = 3)
	Existe coincidência entre o contexto dos cenários envolvidos.	*high_contextSimilarity* (peso = 4)
	Existe coincidência de ao menos um dos atores.	*oneOrMoreActorSimilarity* (peso =3)

Figura 70 - Heurísticas para identificação de relacionamentos

```
PlanLibrary
begin
  Event complement(A,B) is EndEvent
  begin
    composedBy
      high_contextSimilarity(A,B) by context with weight = '1.5';
      high_goalSimilarity(A,B) by goal with weight = '2.0';
      high_actorSimilarity(A,B) by actor with weight = 1.5';
      high_resourceSimilarity(A,B) by resource with weight = '1.0';
      low_episodeSimilarity(A,B) by episode with weight = '4.0';
  end
  Event equivalence(A,B) is EndEvent
  begin
    composedBy
      high_contextSimilarity(A,B) by context with weight = '1.5';
      high_goalSimilarity(A,B) by goal with weight = '2.0';
      high_actorSimilarity(A,B) by actor with weight = '1.5';
      low_resourceSimilarity(A,B) by resource with weight = '1.0';
      high_episodeSimilarity(A,B) by episode with weight = '4.0';
  end
  Event subSet(A,B) is EndEvent
  begin
    composedBy
      subSetContextSimilarity(A,B) by context with weight = '3.0';
      low_goalSimilarity(A,B) by goal with weight = '1.5';
      high_actorSimilarity(A,B) by actor with weight = '2.5';
      low_episodeSimilarity(A,B) by episode with weight = '3.0';
  end
  Event preCondition(A,B) is EndEvent
  begin
    composedBy
      scenarioInsideContext(A,B) by context with weight = '4';
      high_actorSimilarity(A,B) by actor with weight = '3';
      low_episodeSimilarity(A,B) by episode with weight = '3';
  end
  Event possiblePrecedence(A,B) is EndEvent
  begin
    composedBy
      high_contextSimilarity(A,B) by context with weight = '4';
      low_goalSimilarity(A,B) by goal with weight = '3';
      oneOrMoreActorSimilarity(A,B) by actor with weight = '3';
  end
  Event detour(A,B) is EndEvent
  begin
    composedBy
      scenarioInsideContext(A,B) by context with weight = '2';
      high_goalSimilarity(A,B) by goal with weight = '2';
      high_actorSimilarity(A,B) by actor with weight = '2';
      high_resourceSimilarity(A,B) by resource with weight = '2';
      scenarioIsExceptionOfScenario(A,B) by episode with weight = '1';
      ScenarioIsEpisodeOfScenario(A,B) by episode with weight = '1';
  end
  Event exception(A,B) is EndEvent
  begin
    composedBy
      scenarioIsExceptionOfScenario(A,B) by episode with weight = '10';
  end
  Event inclusion(A,B) is EndEvent
  begin
    composedBy
      ScenarioIsEpisodeOfScenario(A,B) by episode with weight = '10';
  end
end
```

Figura 71 - Biblioteca de Planos de Relacionamentos – Parte II: Relacionamentos

A biblioteca de planos foi especificada a partir das heurísticas para identificação dos relacionamentos apresentadas na seção 3.2 e sumarizadas na figura 70. Foi adotado um valor fixo (igual a 10) para ser distribuído entre os

pesos das regras de composição. A distribuição inicial dos pesos foi realizada a partir do grau de importância definida na heurística (Idênticos > Existe grande > Existe > Existe pequena > Pode Existir) e ajustados durante a aplicação da técnica no estudo de caso, de maneira a diminuir a quantidade de identificações erradas, ou seja, as identificações de relacionamentos diferentes daqueles obtidos originalmente no estudo de caso conduzido por Breitman. A figura 71 apresenta parte da biblioteca de planos de relacionamentos utilizadas pelo sistema.

O resultado da aplicação da técnica de reconhecimento de planos utilizando os fatos observados sobre similaridades entre os cenários *malfunction occurs* – Versão I e *malfunction* – Versão I é apresentado a seguir:

```
Recognition
begin
  complement ('malfunction occurs':'1','malfunction':'1') [' 0.45'] {
    goal:high_goalSimilarity('malfunction occurs':'1','malfunction':'1')['1.0']
    context:high_contextSimilarity('malfunction occurs':'1','malfunction':'1')['1.0']
    resource:high_resourceSimilarity('malfunction occurs':'1','malfunction':'1')['1.0']
  }.
  equivalence ('malfunction occurs':'1','malfunction':'1') [' 0.47'] {
    goal:high_goalSimilarity('malfunction occurs':'1','malfunction':'1')['1.0']
    context:high_contextSimilarity('malfunction occurs':'1','malfunction':'1')['1.0']
    episode:high_episodeSimilarity('malfunction occurs':'1','malfunction':'1')['0.3099999']
  }.
  detour ('malfunction occurs':'1','malfunction':'1') [' 0.40'] {
    goal:high_goalSimilarity('malfunction occurs':'1','malfunction':'1')['1.0']
    resource:high_resourceSimilarity('malfunction occurs':'1','malfunction':'1')['1.0']
  }.
  possiblePrecedence ('malfunction occurs':'1','malfunction':'1') [' 0.40'] {
    context:high_contextSimilarity('malfunction occurs':'1','malfunction':'1')['1.0']
  }.
  subSet ('malfunction occurs':'1','malfunction':'1') [' 0.30'] {
    context:subSetContextSimilarity ('malfunction occurs':'1','malfunction':'1')  {
      isa high_contextSimilarity('malfunction occurs':'1','malfunction':'1')['1.0']
    }
  }.
end
```

Este resultado obtido indica que o melhor plano (possui o maior nível de satisfação – 0,47) que pode ser utilizado para explicar os fatos observados sobre similaridades entre os cenários é o que especifica a existência de um relacionamento de equivalência entre eles.

5.1.2. Identificação das Operações

Esta atividade tem como objetivo identificar e armazenar as informações sobre as operações realizadas pelo desenvolvedor que modificaram os cenários de uma configuração criando uma nova configuração. A figura 72 mostra o processo

de identificação das operações que é composto pela identificação das diferenças entre as versões dos cenários (*Find Differences*), pela geração dos transformadores de rastreamento (*HandleDiff*) e pelo reconhecimento dos planos de operações (*Find Plan*). Este processo corresponde ao mecanismo de rastreamento apresentado no capítulo 4 e configurado para o tratamento de artefatos que representem cenários e suas operações.

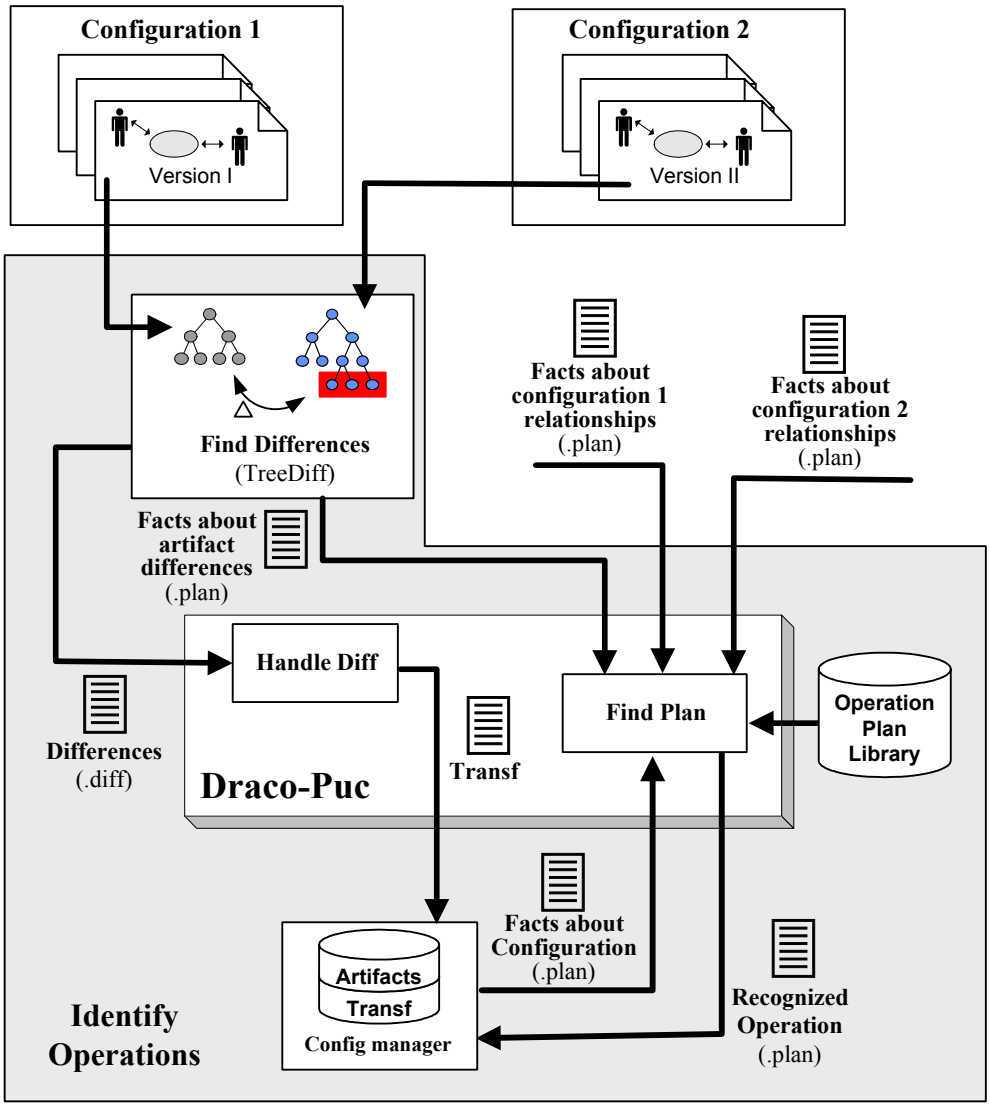

Figura 72 - Processo de identificação das operações entre cenários

A configuração do mecanismo de rastreamento reside na criação da biblioteca de planos correspondente às operações sobre cenários. Os fatos observados desta biblioteca são relativos às diferenças encontradas entre as versões de cenário, as mudanças na configuração do sistema em desenvolvimento; e aos relacionamentos existentes entre os cenários de cada uma das duas configurações em processamento.

Title	malfunction occurs
Goal	how to proceed when a malfunction occurs
Context	4th floor of building 32
Actor	Control system
Actor	4th floor of building 32
Resource	Ceiling light groups
Resource	Control panel
Episodes	inform facility manager of malfunction inform user of malfunction IF Hallway section is controlable neither automatically nor manually THEN Ceiling light groups have to be on Control system supports the facility manager in finding the reasons of malfunction store malfunction facility manager can correct manually malfunction s undetected by the Control system

Figura 73 – Cenário *malfunction occurs* – Versão II

Complementando o exemplo apresentado na seção anterior, onde foram identificados os relacionamentos entre os cenários *malfunction occurs* – Versão I e *malfunction* – Versão I, será agora utilizada a Versão II do cenário *malfunction occurs* (figura 73) e a mudança da configuração do sistema relativa à exclusão do cenário *malfunction* da nova configuração. O objetivo do exemplo é mostrar os passos seguidos na identificação da operação executada sobre estes cenários.

A figura 74 mostra o resultado da aplicação do algoritmo *TreeDiff* utilizado para identificar as mudanças existentes entre a versão I e a versão II do cenário *malfunction occurs*.

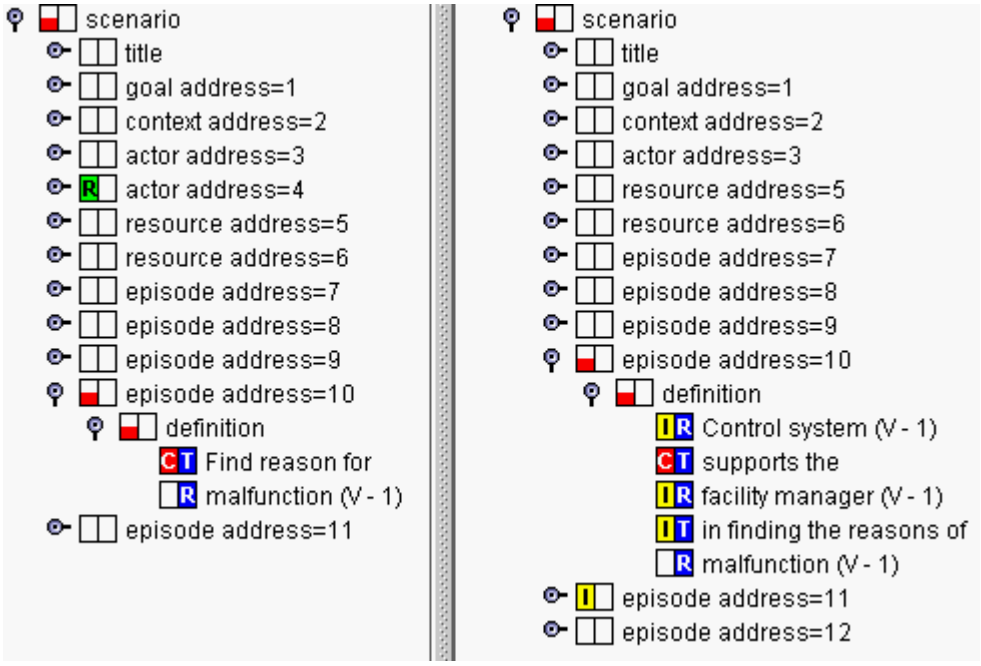

Figura 74 – Resultado da comparação entre as versões I (esquerda) e II (direita) do cenário *malfunction occurs*

O resultado desta comparação (inserção de um episódio) é descrito na forma de um novo fato observado. Os fatos observados para este exemplo são os seguintes:

```
loadLibrary "operationsLibrary.plan" .
Observations
Begin
// fato observado na atividade de identificação de relacionamentos
    equivalence ('malfunction occurs':1,'malfunction':1) ['0.47']

// fatos gerados pelo gerente de configuração
    scenarioRemoved ('malfunction':1) ['1.0']
    episodeRemoved ('malfunction':1 , 'e1') ['1.0']

// fato gerado pela aplicação do TreeDiff
    episodeAdded ('malfunction occurs':1 , 'e1') ['1.0']
end
```

Um exemplo da biblioteca de planos de operações é mostrada a seguir. A descrição completa da biblioteca encontra-se no apêndice E.

```
PlanLibrary
begin
/******************  Observed events  ******************/
/* Relationships */
    Event complement(A,B);
    Event equivalence(A,B);
    Event subSet(A,B);
    Event inclusion(A,B);
/* Configuration Manager */
    Event scenarioRemoved(A);
    Event scenarioAdded(A);
/* Episode changes */
    Event episodeRemoved(A,B)  /* A: scenario  B: episode */
        begin
          isa episodeModification
        end
    Event episodeAdded(A,B)  /* A: scenario  B: episode */
        begin
          isa episodeModification
        end
    Event episodeContentChanged(A,B)
        begin
          isa episodeModification
        end
/******************  Intermediate events  ******************/
    Event episodeModification(A,B);
    Event episodeMoved(A,B,C) /* A: firstScenario  B: secondScenario   C: episode */
        begin
          composedBy
            episodeRemoved(A,C) by source with weight = '1.0';
            episodeAdded(B,C) by target with weight = '1.0';
        end
/******************  Final events  ******************/
    Event fusionOperation(A,B) is EndEvent;  /* B foi unido em A */
    Event fusionOperationInOldScen(A,B)   /* B foi unido em A */
        begin
            isa fusionOperation
          composedBy
```

```
            scenarioRemoved(B) by r1 with weight = '1.0';
            complement(A-,B) by r2 with weight = '1.0';
            episodeMoved(B,A,*) by r3 with weight = '1.0';
        end
    Event fusionOperationInNewScen(A,B,C) /* B foi unido em A no novo cenario C */
        begin
            isa fusionOperation
          composedBy
            scenarioRemoved(A) by r1 with weight = '1.0';
            scenarioRemoved(B) by r2 with weight = '1.0';
            scenarioAdded(C) by r3 with weight = '1.0';
            complement(A,B) by r4 with weight = '1.0';
            episodeMoved(B,C,*) by r5 with weight = '1.0';
            episodeMoved(A,C,*) by r6 with weight = '1.0';
        end
    Event encapsulationOperation(A,B) is EndEvent  /* B foi encapsulado em A */
        begin
          composedBy
            scenarioRemoved(B) by r1 with weight = '1.0';
            equivalence(A-,B) by r2 with weight = '1.0';
            episodeMoved(B,A,*) by r3 with weight = '1.0';
        end
end
```


O resultado da aplicação da técnica de reconhecimento de planos de operações utilizando o conjunto de fatos observados é mostrado a seguir, permitindo que se conclua que o cenário *malfunction* – Versão I foi encapsulado no *malfunction occurs* – Versão II.

```
Recognition
begin
  encapsulationOperation ('malfunction occurs'1,'malfunction'1) [' 0.82'] {
    r1:scenarioRemoved('malfunction':1)['1.0']
    r2:equivalence('malfunction occurs':1,'malfunction':1)['0.47']
    r3:episodeMoved ('malfunction'1,*,'e1')  {
      source:episodeRemoved('malfunction':1,'e1')['1.0']
      target:episodeAdded('malfunction occurs':1,'e1')['1.0']
    }
  }.
end
```

5.2. Protótipo da Ferramenta de Suporte ao Processo

Foi criado um protótipo de ferramenta de suporte ao processo de rastreamento com o objetivo de facilitar sua aplicação no estudo de caso. Esta ferramenta (*DiffTraceTool*) foi desenvolvida no paradigma de orientação a objetos e sua implementação foi feita em Java. As principais funcionalidades disponíveis na ferramenta são:

a) Realiza o controle de versões do sistema através da utilização das classes mostradas no apêndice D. A figura 75 mostra um exemplo da tela principal da

ferramenta. A marca 1 mostra as diversas configurações do sistema. As versões dos artefatos correspondentes a configuração selecionada em 1 são mostradas na janela marcada com o número 2.

b) Fornece suporte a importação dos cenários armazenados na base de dados utilizada no desenvolvimento original do estudo de caso. As informações inicialmente disponíveis sobre o sistema utilizado como estudo de caso estavam armazenadas de forma desestruturada (formato HTML) em um banco de dados Access, e tiveram que ser portadas para o formato especificado na seção 4.1. ;

c) Aplicação do algoritmo *TreeDiff* e visualização das diferenças e similaridades entre os artefatos. As diferenças são mostradas graficamente (marca 3 da figura 75) e descritas textualmente (marca 4 da figura 75).

d) Reconhecimento de relacionamentos e operações entre cenários através da integração entre o algoritmo TreeDiff e a Máquina Draco-PUC;

e) Ferramenta para edição de cenários. Além das operações normais sobre árvores (inserção, remoção e renomeação de nós) esta ferramenta permite a inclusão automática de referências ao LAL, ou seja, quando da inserção de um texto verifica se existe no LAL algum símbolo relacionado a este texto. A figura 76 mostra um exemplo de edição de um cenário onde: a marca 1 indica o cenário em edição; a marca 2 mostra os termos do LAL que correspondem ao texto do nó selecionado em 1; e a marca 3 apresenta as versões existentes para o símbolo selecionado em 2.

f) Geração dos arquivos em HTML para cada artefato do sistema através da aplicação das transformações XSL correspondentes. A figura 77 mostra um exemplo dos arquivos gerados. O painel da esquerda mostra as versões de uma configuração. A versão selecionada é mostrada no painel superior direito. Cada vez que uma referência a um símbolo do LAL for selecionada a descrição do símbolo é mostrada no painel inferior direito.

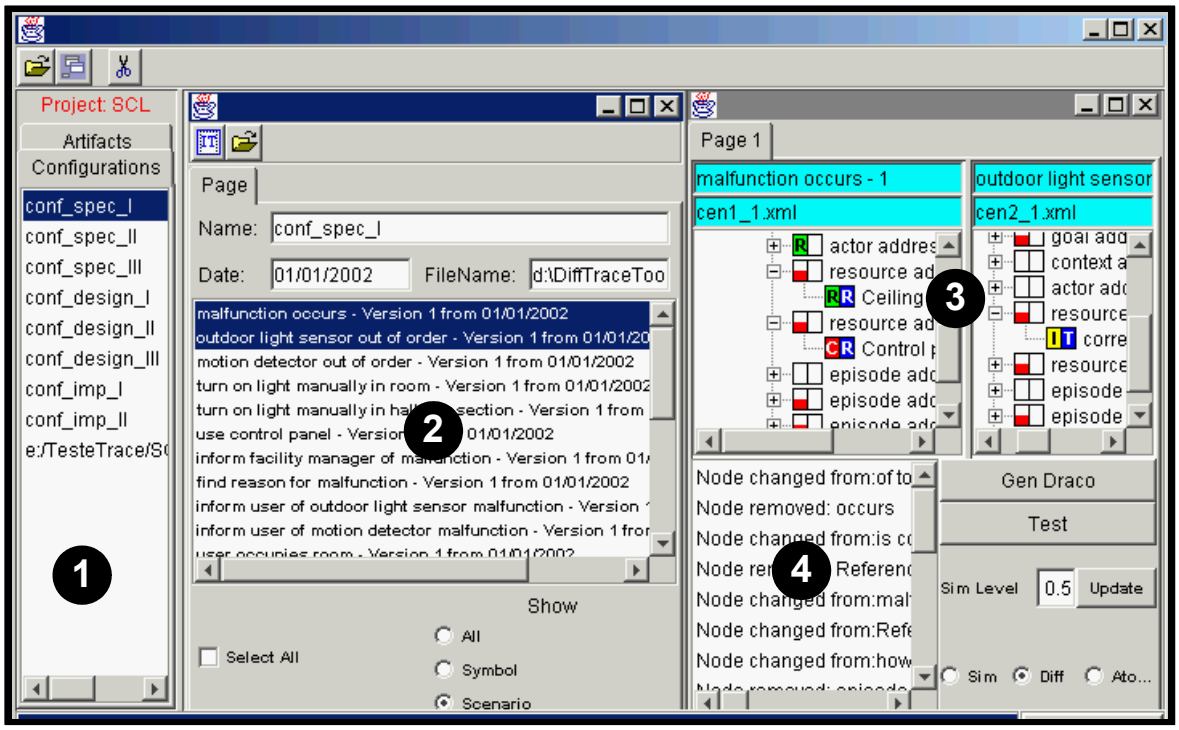

Figura 75 – Tela principal da ferramenta *DiffTraceTool*

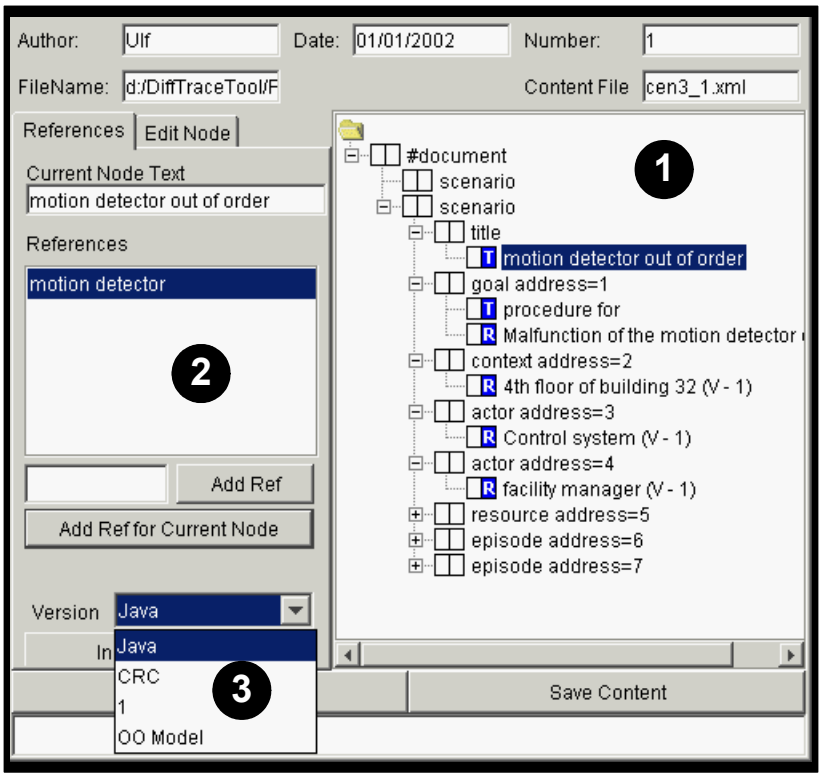

Figura 76 – Tela de edição de cenários

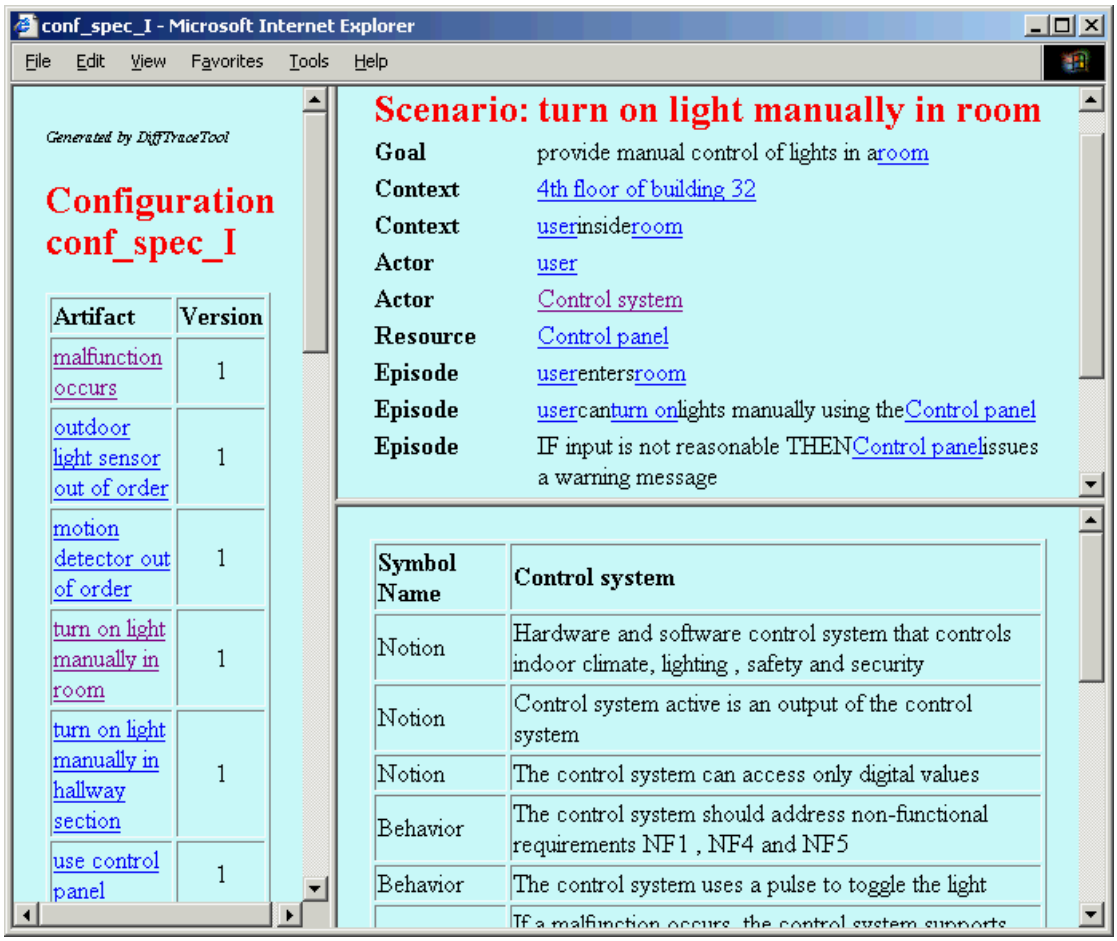

Figura 77 – Exemplos dos arquivos HTML gerados

5.3. Estudo de Caso

O sistema escolhido para este estudo de caso é o que foi utilizado na tese de Breitman [10] na definição de sua taxonomia para a evolução dos cenários. Este sistema faz a simulação do controle da iluminação do Departamento de Informática da Universidade de Kaiserslautern e foi proposto inicialmente como estudo de caso em um workshop de engenharia de requisitos realizado no Schloss Daghstuhl, Alemanha em 1999 [75]. A descrição completa do problema tratado pode ser encontrada em http://rn.informatik.uni-kl.de/~recs/problem. O motivo da escolha deste sistema para ser utilizado como estudo de caso é a existência de todo o histórico do seu desenvolvimento, bem como dos relacionamentos e operações entre cenários, e desta maneira poder ser utilizado na confrontação com os relacionamentos e operações identificados através do sistema descrito neste capítulo. Este histórico é composto por 8 configurações mostradas na figura 78 e

encontra-se disponível no seguinte no endereço http://stones.les.inf.puc-rio.br/Karin/exemplo/index.html.

Conf.	Descrição	Cen.	Relac.	Oper.
Spec I	Primeira configuração dos cenários da base. Derivada a partir do Léxico Ampliado da Linguagem desta aplicação através de heurísticas próprias.	23	21	23
Spec II	Segunda configuração do conjunto de cenários. Resultado da remoção de redundâncias e reorganização	17	16	9
Spec III	Terceira configuração do conjunto de cenários. Resultado do aumento da compreensão da descrição do problema e do refinamento das informações.	10	14	5
Design I	Primeira configuração dos cenários de desenho. Adaptados da configuração anterior e atualizados de modo a refletir a modelagem dos cartões CRC.	10	14	10
Design II	Segunda configuração dos cenários de desenho. Refletem o modelo orientado a objetos (OO) derivado a partir dos cartões CRC.	11	15	7
Design III	Última configuração dos cenários de desenho. Reflete o corte no modelo OO e representa as interações entre o sistema a ser construído e entidades externas.	10	11	7
Imp I	Primeira configuração dos cenários de implementação. Reflete os ajustes realizados no modelo OO e incorpora as classes de interface que tiveram de ser adicionadas.	7	4	5
Imp II	Esta configuração descreve os cenários do ponto de vista externo (do cliente). Representa a interação dos usuários do sistema com sua implementação.	7	5	7

Figura 78 – Informações disponíveis sobre o sistema utilizado no estudo de caso.

5.3.1. Exemplo

Será apresentado um exemplo completo da utilização do sistema de rastreamento. Serão utilizados neste exemplo os cenários *define light scene* – Versão I (figura 79), *user defines light scene* – Versão I (figura 80) e *facility manager defines light scene* – Versão I (figura 81). O primeiro cenário existia na primeira configuração (*Spec I*) e foi substituído pelos dois últimos cenários na segunda configuração (*Spec II*). O objetivo deste exemplo é mostrar a utilização do mecanismo de rastreamento na identificação das operações realizadas sobre os cenários da configuração *Spec I* de maneira a obter os cenários existentes na configuração *Spec II* .

Identificação de Relacionamentos

Esta atividade tem como objetivo encontrar os relacionamentos existentes entre os cenários de uma mesma configuração. Neste exemplo, procuraremos

identificar os relacionamentos existentes entre os cenários da configuração *Spec II - user defines light scene* e *facility manager defines light scene*.

O primeiro passo desta atividade é a procura pelas similaridades existentes entre estes cenários. Para tanto, inicialmente é aplicado o algoritmo *TreeDiff*, sendo identificadas as diferenças mostradas na figura 82.

Scenario: define light scene

Goal	define user preferred light scene
Context	4th floor of building 32
Context	user in room
Actor	user
Actor	Control system
Resource	Default light scene value
Resource	Control panel
Episode	user places himself/herself near Control panel
Episode	user retrieves Default light scene
Episode	user changes desired values for his ChooseLightScene
Episode	IF input is not reasonable THEN Control system issues a warning

Figura 79 – Cenário *define light scene* – Versão I

Scenario: user defines light scene

Goal	define user preferred light scene
Context	4th floor of building 32
Context	user in room
Actor	user
Actor	Control system
Resource	Default light scene value
Resource	Control panel
Episode	user places himself/herself near Control panel
Episode	user retrieves Default light scene
Episode	user changes desired values for his/hers ChooseLightScene
Episode	IF input is not reasonable THEN Control system issues a warning
Episode	Control system implements ChooseLightScene
Episode	EXCEPTION: IF outdoor light sensor malfuction THEN Control system establishes Default light scene in room

Figura 80 – Cenário *user defines light scene* – Versão I

Scenario: facility manager defines light scene

Goal	define facility manager preferred light scene
Context	4th floor of building 32
Context	facility manager in room or Hallway section
Actor	facility manager
Actor	Control system
Resource	Default light scene value
Resource	Control panel
Episode	facility manager places himself/herself near Control panel
Episode	facility manager retrieves Default light scene
Episode	facility manager changes desired values for his/hers ChooseLightScene
Episode	IF input is not reasonable THEN Control system issues a warning
Episode	Control system implements ChooseLightScene
Episode	EXCEPTION: IF outdoor light sensor malfuction THEN Control system establishes all ceiling group lights on in Hallway section
Episode	EXCEPTION: IF outdoor light sensor malfuction THEN Control system establishes Default light scene in room

Figura 81 - Cenário *facility manager defines light scene* – Versão I

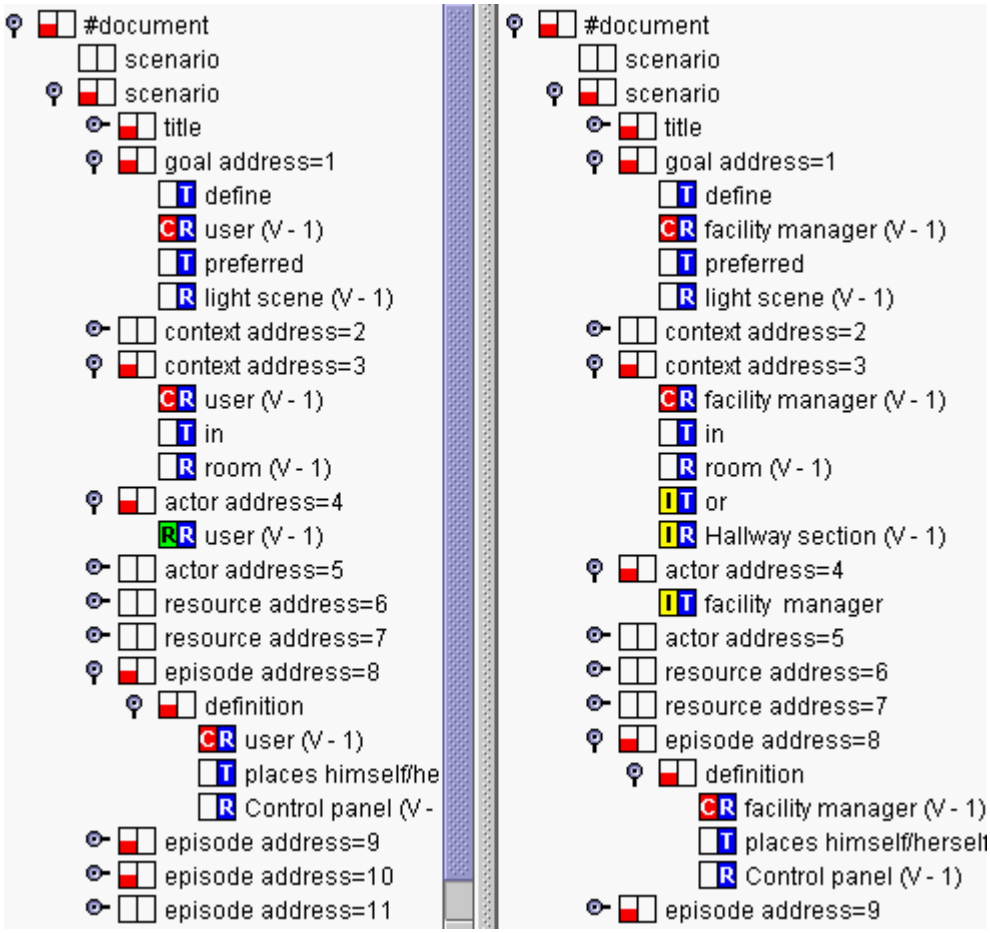

Figura 82 – Resultado da comparação entre os cenários *user defines light scene* – Versão I (esquerda) e *facility manager defines light scene* – Versão I (direita)

A partir das diferenças identificadas são gerados os fatos observados sobre as similaridades existentes entre os elementos dos cenários. A figura 83 apresenta a descrição dos fatos observados para os cenários do exemplo, que mostram a existência de uma alta similaridade de objetivos (0,56), de contexto (0,71), de episódios (0,71) e de recursos (1,0) entre os cenários analisados.

```
loadLibrary "relationshipLibrary.plan" .
Observations
begin
    high_goalSimilarity('user defines light scene':1 , 'facility manager defines light scene':1) ['0.56']
    high_contextSimilarity('user defines light scene':1 , 'facility manager defines light scene':1) ['0.71']
    high_episodeSimilarity('user defines light scene':1 , 'facility manager defines light scene':1) ['0.71']
    high_resourceSimilarity('user defines light scene':1 , 'facility manager defines light scene':1) ['1.0']
end
```

Figura 83 – Fatos observados sobre as similaridades existentes entre os cenários *user defines light scene* – Versão I e *facility manager defines light scene* – Versão I

Os fatos observados são então utilizados pela técnica de reconhecimento de planos que resulta na lista de planos reconhecidos mostrados na figura 84. O plano marcado nesta figura foi o que obteve o maior grau de satisfação (0,50), permitindo que se identifique a existência do relacionamento de equivalência entre os cenários cenários *user defines light scene* e *facility manager defines light scene*.

A segunda atividade corresponde a identificação das operações entre os cenários. A figura 85 mostra os fatos observados utilizados pela técnica de reconhecimento de planos de operações. Os fatos observados de números 1 a 3 foram gerados pelo Gerente de Configuração e correspondem a remoção do cenário original e a adição dos dois novos cenários. Os fatos 4 a 11 referem-se a remoção dos episódios do cenário original e a adição de episódios nos novos cenários. O fato 12 corresponde ao relacionamento identificado na atividade anterior.

```
Recognition begin
  complement ('user defines light scene'1,'facility manager defines light scene'1) [' 0.32']{
    goal:high_goalSimilarity('user defines light scene':1,'facility manager defines light scene':1)['0.56249994']
    context:high_contextSimilarity('user defines light scene':1,'facility manager defines light scene':1)['0.7083333']
    resource:high_resourceSimilarity('user defines light scene':1,'facility manager defines light scene':1)['1.0']
  }.
  equivalence ('user defines light scene'1,'facility manager defines light scene'1) [' 0.50']{
    goal:high_goalSimilarity('user defines light scene':1,'facility manager defines light scene':1)['0.56249994']
    context:high_contextSimilarity('user defines light scene':1,'facility manager defines light scene':1)['0.7083333']
    episode:high_episodeSimilarity('user defines light scene':1,'facility manager defines light scene':1)['0.70833343']
  }.
  detour ('user defines light scene'1,'facility manager defines light scene'1) [' 0.31'] {
    goal:high_goalSimilarity('user defines light scene':1,'facility manager defines light scene':1)['0.56249994']
    resource:high_resourceSimilarity('user defines light scene':1,'facility manager defines light scene':1)['1.0']
  }.
  possiblePrecedence ('user defines light scene'1,'facility manager defines light scene'1) [' 0.28'] {
    context:high_contextSimilarity('user defines light scene':1,'facility manager defines light scene':1)['0.7083333']
  }.
  subSet ('user defines light scene'1,'facility manager defines light scene'1) [' 0.21'] {
    context:subSetContextSimilarity ('user defines light scene':1,'facility manager defines light scene':1)  {
      isa high_contextSimilarity('user defines light scene':1,'facility manager defines light scene':1)['0.7083333']
    }
  }.
end
```

Figura 84 – Resultado do reconhecimento de planos de relacionamentos entre os cenários *user defines light scene* – Versão I e *facility manager defines light scene* – Versão I

```
loadLibrary "operationsLibrary.plan" .
Observations begin
        scenarioRemoved('define light scene':1) ['1.0']                          // Fato 1
        scenarioAdded('user defines light scene':1) ['1.0']                       // Fato 2
        scenarioAdded('facility manager defines light scene':1) ['1.0']           // Fato 3
        episodeRemoved ('define light scene':1 , 'e1') ['1.0']                    // Fato 4
        episodeAdded ('user defines light scene':1 , 'e1') ['1.0']                // Fato 5
        episodeRemoved ('define light scene':1 , 'e2') ['1.0']                    // Fato 6
        episodeAdded ('user defines light scene':1 , 'e2') ['1.0']                // Fato 7
        episodeRemoved ('define light scene':1 , 'e3') ['1.0']                    // Fato 8
        episodeRemoved ('define light scene':1 , 'e4') ['1.0']                    // Fato 9
        episodeAdded ('user defines light scene':1 , 'e4') ['1.0']                // Fato 10
        episodeAdded ('facility manager defines light scene':1 , 'e4') ['1.0']    // Fato 11
        equivalence('user defines light scene':1,'facility manager defines light scene':1) ['0.32']   // Fato 12
end
```

Figura 85 – Fatos observados sobre os cenários *define light scene* – Versão I , *user defines light scene* – Versão I e *facility manager defines light scene* – Versão I a serem utilizados pela técnica de reconhecimento de planos de operações.

Os fatos observados da figura 85 são utilizados pela técnica de reconhecimento de planos para identificar qual a operação que pode explicar este conjunto de fatos. A figura 86 mostra o resultado da aplicação da técnica. O plano identificado fornece um grau de satisfação de 0,89 para a operação de especialização entre os cenários do exemplo.

```
Recognition
begin
  specializationOperation ('define light scene':1,'user defines light scene':1,'facility
manager defines light scene':1) [' 0.89'] {
    r1:scenarioRemoved('define light scene':1)['1.0']
    r2:scenarioAdded('user defines light scene':1)['1.0']
    r3:scenarioAdded('facility manager defines light scene':1)['1.0']
    r4:equivalence('user defines light scene':1,'facility manager defines light
scene':1)['0.50']
    r5:episodeMoved ('define light scene':1,'user defines light scene':1,'e1') {
      source:episodeRemoved('define light scene':1,'e1')['1.0']
      target:episodeAdded('user defines light scene':1,'e1')['1.0']
    }
    r6:episodeMoved ('define light scene':1,'facility manager defines light scene':1,'e1') {
      source:episodeRemoved('define light scene':1,'e1')['1.0']
      target:episodeAdded('facility manager defines light scene':1,'e1')['1.0']
    }
  }.
end
```

Figura 86 – Resultado da técnica de reconhecimento de planos de operações

A figura XXX apresenta os resultados obtidos pela aplicação do mecanismo de rastreamento sobre o exemplo apresentado.

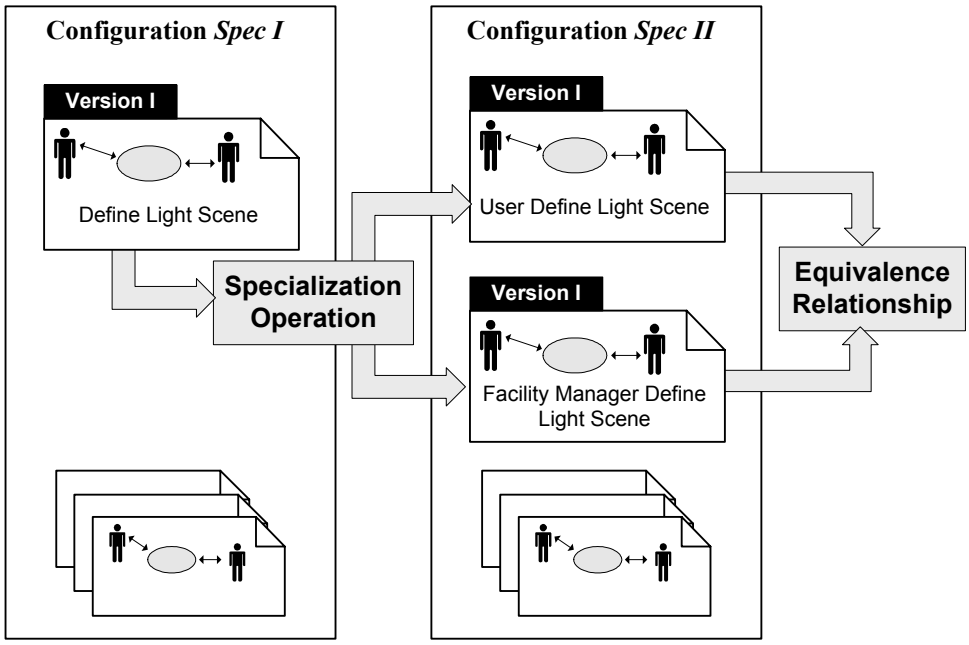

Figura 87 – Resultados da aplicação do mecanismo de rastreamento sobre o exemplo

5.3.2. Resultados Obtidos

Os resultados obtidos a partir da execução do estudo de caso utilizando o sistema de rastreamento apresentado neste capítulo foram os seguintes:

1. Quanto aos transformadores de rastreamento gerados a partir da comparação entre duas versões de um artefato:
   1.1. Foram gerados todos os transformadores de rastreamento. Estes transformadores são baseados unicamente nas diferenças existentes entre as versões analisadas.
   1.2. A aplicação dos transformadores sobre uma versão teve como resultado a versão seguinte do artefato. Este fato foi comprovado através da aplicação do algoritmo *TreeDiff* entre o artefato que foi utilizado na geração da transformação e o artefato gerado pela aplicação da transformação.

   Conclusão: É possível capturar todas as diferenças representativas das ações realizadas pelo desenvolvedor na forma de transformações a serem aplicadas sobre a versão inicial de um artefato de maneira a obter sua nova versão.
2. Quanto à técnica de reconhecimento de planos:
   2.1. Foi notada uma baixa eficiência na implementação do reconhecimento de planos através de transformações.
   2.2. A linguagem utilizada para descrever os planos é adequada, permitindo que se capture todas as heurísticas necessárias à identificação das operações realizadas pelo desenvolvedor.
   2.3. A abordagem utilizada para quantificar o nível de satisfação de um plano se mostrou adequado para ser utilizado como critério de seleção dos planos a serem reconhecidos.

   Conclusão: A implementação da técnica de reconhecimento de planos através de transformações não possui a eficiência necessária para tratar sistemas grandes. A linguagem de descrição de planos é adequada.
3. Quanto ao reconhecimento dos relacionamentos:
   3.1. Ocorreu um número razoável de resultados falsos positivos, ou seja, o sistema identificou um relacionamento errado. O principal motivo que deu origem a estes erros foi o fato de que as heurísticas não permitem identificar com certeza qual o relacionamento existente.

Conclusão: os fatos necessários a identificação dos relacionamentos podem ser inferidos a partir das similaridades existentes entre os elementos dos cenários. As heurísticas para identificação dos relacionamentos não tem a precisão adequada.

4. Quanto ao reconhecimento das operações:
   4.1. A quantidade de fatos gerados a serem tratados pelo reconhecimento de planos é muito grande.
   4.2. A totalidade das operações intra-cenários foi identificada corretamente.
   4.3. A maior parte das operações inter-cenários foi identificada corretamente ocorrendo porém alguns resultados falso positivos.
   4.4. A principal causa de erro na identificação das operações foi devido ao grande número de ações realizadas entre duas versões dos artefatos, quando algumas ações foram atribuídas a outras operações.
   4.5. Uma causa de erro secundária foi devido às heurísticas utilizadas não poderem precisar exatamente qual a operação a ser reconhecida.

   Conclusão: A identificação das operações possui uma boa precisão mas precisam ser melhorados os aspectos relacionados a sua eficiência.

5.4. Conclusão

Conforme apresentado na introdução deste capítulo, um sistema de rastreamento de requisitos deve ser capaz de auxiliar na documentação, entendimento e gerenciamento da evolução dos requisitos. O sistema apresentado neste capítulo possui como características principais: o armazenamento dos transformadores que capturam as ações realizadas pelo desenvolvedor na evolução do sistema; a possibilidade de aplicar estes transformadores e desta maneira replicar as ações do desenvolvedor sobre os artefatos; e a identificação e armazenamento da operação que caracteriza estas ações. Estas características facilitam a documentação, pois todas as informações necessárias podem ser capturadas de maneira semi-automática, manipuladas de forma a gerar qualquer tipo de documentação desejada, inclusive pela própria aplicação de outras transformações. Quanto ao entendimento dos requisitos, a obtenção das operações sobre cenários permite que se visualize o conjunto de ações correspondentes em um grau de abstração maior, além de se poder criar outros dispositivos de

visualização, como por exemplo um dispositivo que aplica os transformadores gerados para fornecer a visualização da evolução de um requisito, nos moldes dos sistemas de animação da execução do código fonte. Os transformadores gerados podem ser ainda utilizados na geração de uma série de informações úteis ao gerenciamento dos requisitos, tais como a verificação de dependências entre os artefatos e o impacto das modificações. A geração destas informações pode ser realizada através da aplicação de novas transformações sobre o conjunto formado pelas transformações de rastreamento e pelos planos que foram reconhecidos, identificando, por exemplo, as referências existentes entre os artefatos, inferindo sobre a dependência existente entre eles; as mudanças da configuração do sistema, inferindo os artefatos que foram reunidos em um só; e o conjunto de operações que foram aplicados sobre um artefato até que o produto final do sistema tenha sido obtido, concluindo sobre o impacto das mudanças.

Quanto aos resultados obtidos pelo estudo de caso (seção 5.3.2), algumas considerações devem ser feitas quanto a eficiência e a eficácia do sistema.

Considerações sobre a eficiência

A utilização do algoritmo *TreeDiff* se mostrou adequada e eficiente, podendo ser utilizada para a identificação das diferenças existentes entre dois artefatos quaisquer descritos em XML. A utilização do paradigma transformacional na implementação do processo de reconhecimento de planos se mostrou pouco eficiente, não sendo adequada para tratar sistemas grandes onde o número de fatos observados é elevado (apontado pelos itens 2.1 e 4.1).

Duas alternativas podem ser utilizadas para resolver o problema da pouca eficiência do reconhecimento de planos. A primeira mantendo o enfoque transformacional e otimizando aspectos específicos do transformador, e a segunda abandonando o enfoque transformacional e adotando uma abordagem mais eficiente.

Adotando-se a primeira alternativa, ou seja, mantendo-se o enfoque transformacional no reconhecimento de planos, algumas otimizações que podem ser realizadas no processo de reconhecimento envolvem a manutenção de uma lista indexada entre os eventos possíveis de serem observados e os diversos planos que os utilizam; a realização de escolhas intermediárias de maneira a descartar os planos que no final da execução tem menor probabilidade de serem escolhidos; e a

melhorara do cálculo do valor para os eventos inferidos a partir dos fatos observados.

Para a segunda alternativa, o enfoque transformacional para o reconhecimento de planos poderia ser substituído por algoritmos que utilizam técnicas baseadas em inferências diretas ou na teoria de grafos, tais como os propostos por Lesh & Etzioni [76] ou Lin & Goebel [77]. Neste caso, devem ser mantidas a mesma linguagem para descrição dos planos atualmente utilizada e o mesmo mecanismo de seleção de planos baseado no seu nível de satisfação.

Outros aspectos podem ser tratados de maneira a aumentar a eficiência do sistema como um todo. Na identificação das operações realizadas sobre uma configuração procuramos identificar todos os relacionamentos existentes entre os diversos cenários da configuração analisada e da próxima configuração (ocasionando o fato apontado no item 4.1). Desta forma são analisados todos os pares de cenários obtidos a partir da combinação das duas configurações do sistema. Por exemplo, a existência de Y cenários em uma configuração e de Z cenários na configuração seguinte, obriga que se pesquisem os relacionamentos que podem existir entre YxZ pares de cenários. Para o objetivo específico do mecanismo de rastreamento em questão, só interessa realizar esta procura entre os pares de cenários onde pelo menos um deles foi inserido ou removido de uma configuração em relação à outra, pois a informação sobre os relacionamentos só é utilizada como heurística para operações inter cenários onde ocorreram modificações na configuração do sistema. Foi mantida esta procura mais ampla no mecanismo pois acreditamos que as informações sobre os diversos tipos de relacionamentos são importantes para o desenvolvedor.

Considerações sobre a eficácia

Existe a necessidade de refinar e estender o conjunto de heurísticas utilizadas na identificação dos relacionamentos e operações (apontado pelos itens 4.3 e 4.5). Para o conjunto de heurísticas utilizadas atualmente torna-se necessária a intervenção do usuário do sistema para escolher, a partir de uma lista priorizada de planos reconhecidos, qual o plano que realmente pode explicar a situação em análise.

Uma questão que fica em aberto é sobre a freqüência com que se deve realizar a criação de novas configurações e sua submissão ao sistema de rastreamento para que sejam capturadas as informações. Se a freqüência for muito

grande ocorrerá uma sobrecarga de processamento e se for muito pequena a quantidade de falsos positivos irá aumentar (apontado pelo item 4.5). Novos estudos, conduzidos ao longo do processo de desenvolvimento de outros sistemas, podem apontar alguns parâmetros a serem utilizados na definição desta freqüência. Acreditamos que esta definição deva ser configurável pelo gerente de desenvolvimento, podendo ser modificada ao longo do processo de desenvolvimento levando em consideração a eficiência e a eficácia na aplicação do mecanismo.

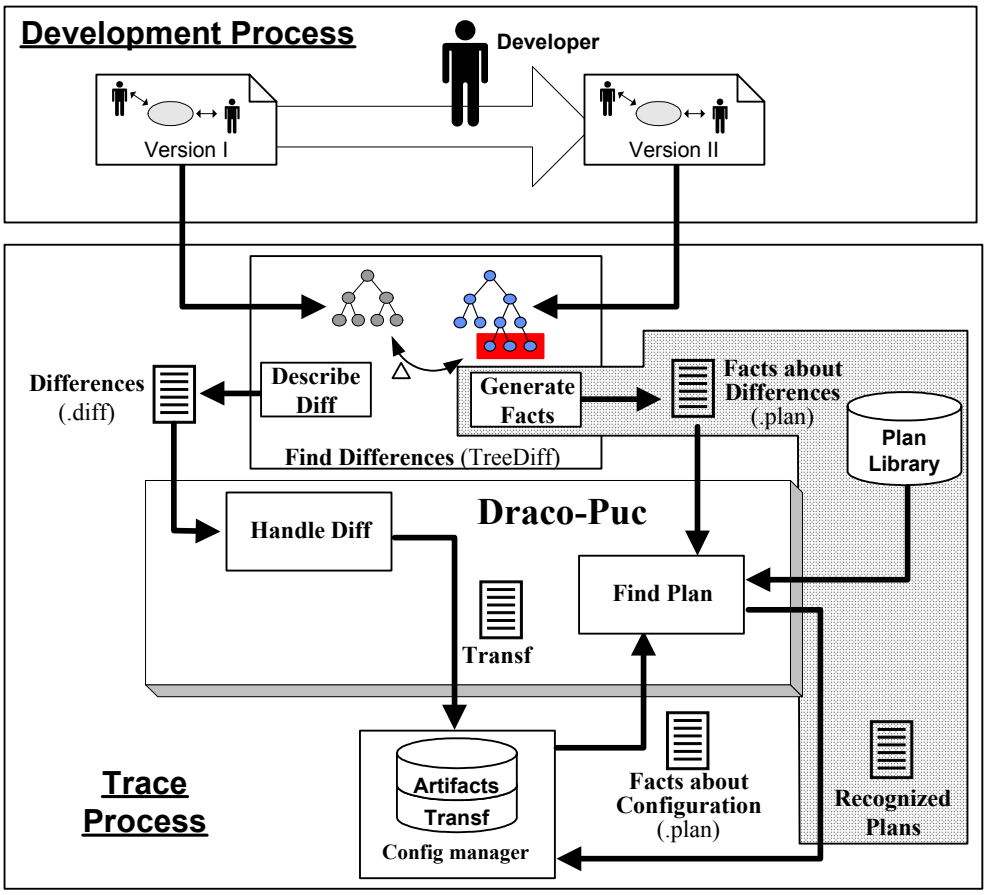

Figura 88 – Atividades e produtos do mecanismo de rastreamento dependentes do tipo de artefato

Neste capítulo o mecanismo foi utilizado na captura de informações de rastreamento relacionadas à evolução de cenários. A utilização do mecanismo em outros artefatos depende da definição do conjunto de operações possíveis de serem realizadas sobre cada tipo de artefato, nos moldes da taxonomia para evolução de cenários apresentada na seção 3.2. A figura 88 destaca as atividades e produtos do mecanismo proposto que são dependentes do tipo do artefato. Para cada tipo de artefato deve ser criada uma biblioteca de planos correspondente, função do conjunto de operações específicas de cada artefato, e os fatos

observados a serem gerados devem ser aqueles relacionados a cada biblioteca. As demais atividades permanecem inalteradas, podendo ser aplicadas diretamente a qualquer tipo de artefato.