

Suporte Automatizado à Gerência da Evolução de Cenários*

Karin Koogan Breitman, Julio Cesar Sampaio do Prado Leite

karin, julio@inf.puc-rio.br

Departamento de Informática - PUC-Rio

Rua Marquês de São Vicente 225 Gávea

Rio de Janeiro – Brasil

Resumo

Neste artigo discutimos aspectos relacionados à evolução de especificações baseadas em cenários. Apresentamos resultados de estudos realizados nesta área, traduzidos em um conjunto de operações. A partir destes resultados tecemos comentários sobre o apoio automatizado a especificações baseadas em cenários.

Abstract

In this article we present some evolutionary aspects of scenarios. Supported by case study results and based on a framework for scenario evolution, we discuss the requirements for a tool that provides automated support for scenario evolution management.

1. Introdução

Um grande passo no processo de definição dos requisitos de uma aplicação reside no reconhecimento dos atores, entidades e ações que descrevem o macrosistema do qual esta aplicação fará parte. Trabalhos recentes conduzidos por nosso grupo de pesquisa exploram a utilização de cenários como técnica de elicitação de requisitos do macrosistema [Leite 97] [Hada97] [Breitman98]. Cenários vem surgindo como opção para a descrição de situações do mundo real que envolvem atores interagindo dentro de um determinado contexto [Zorman95]. Esta interação é descrita através de ações enumeradas em um ou mais episódios. Cenários utilizam elementos conhecidos pelos usuários, facilitando tanto o processo de elicitação de requisitos quanto a validação dos mesmos.

Resultados da nossa prática têm apontado uma crescente necessidade de ferramental para apoio à criação e evolução de especificações baseadas em cenários. Existem na literatura várias propostas de ferramentas que apoiam a criação dos mesmos [Zorman95, Maiden98, Antonelli98]. Todas ferramentas oferecem, em variados graus, apoio as atividades de entrada de dados, criação de conexões (ou elos) entre cenários e facilitam a verificação da consistência do conjunto. Estas atividades são fundamentais na criação da primeira versão dos cenários de uma aplicação porém, à medida em que esta evolui, outras necessidades emergem. Entendemos como evolução o processo de transformação sofrido por uma especificação ao longo da vida útil da aplicação que esta corresponde. Desta forma, uma especificação evolui durante as fases de desenvolvimento e manutenção da aplicação. Nenhuma das ferramentas citadas acima oferece apoio satisfatório à evolução das especificações baseadas em cenários.

Durante o processo evolutivo de especificações em cenários fomos capazes de observar uma série de fenômenos [Breitman98]. Por um lado existem os fenômenos relacionados aos processos a serem automatizados em si, a maneira com que estes são realizados durante a fase inicial de elicitação dos requisitos do sistema e como se deseja que o sistema os automatize. Estes fenômenos têm um caráter transformacional, uma vez que a automatização de processos manuais (ou realizados por um outro sistema) sempre acarreta em mudanças no modo em que estes são conduzidos. Em outra dimensão existem fenômenos relacionados ao amadurecimento do conhecimento dos processos como um todo. É fato que os requisitos de um sistema mudam ao longo do tempo [Lehman91]. Várias causas são atribuídas a este fenômeno, uma das quais o fato dos próprios usuários não conhecerem a totalidade dos requisitos do sistema no momento de sua elicitação. Nossa experiência com elicitação de requisitos em geral, e utilização de cenários em particular, mostra que este fato tem um papel muito importante do ponto de vista da rastreabilidade do sistema.

Neste artigo discutiremos alguns dos aspectos relacionados à evolução de especificações em cenários colocados acima. Parte desta discussão pode ser encontrada em [Breitman98] porém daremos ênfase a necessidade de apoio automatizado que suporte estas tarefas. Desta forma tentaremos discutir os requisitos básicos para uma ferramenta de suporte automatizado à evolução de cenários. O restante deste artigo está dividido da seguinte forma: na próxima seção apresentamos a definição de cenários utilizada; na seção 3 apresentamos os resultados de nossas pesquisas; na seção 4 apresentamos uma discussão sobre o apoio automatizado à gerência da evolução de cenários; na conclusão fazemos um apinhado da discussão deste artigo.

2. Cenários

No contexto da nossa pesquisa utilizamos cenários conforme proposto por Leite em [Leite97]. Central a esta proposta são as seguintes premissas:

- Um cenário descreve situações do macrosistema e suas relações com o sistema.
- Um cenário evolui durante o processo de desenvolvimento do software.
- Os cenários estão naturalmente relacionados como o Léxico Ampliado da Linguagem (LAL)*.
- Cenários utilizam linguagem natural

O modelo de cenários utilizado faz parte de uma proposta maior, o *Requirements Baseline* [Leite97] e tem como objetivos modelar aspectos comportamentais dos requisitos. A estrutura de um cenário está ilustrada na Figura 1 a seguir. Descreveremos cada um dos elementos separadamente.

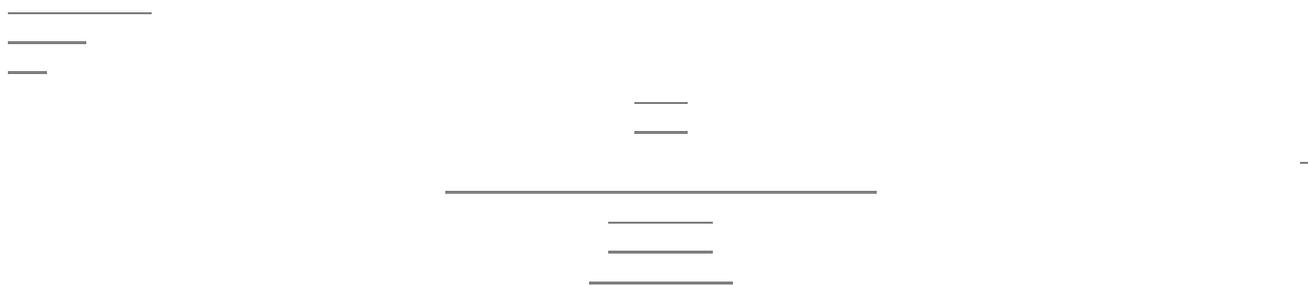


Figura 1 – Diagrama de entidade e relacionamento da estrutura de um cenário

Objetivo: estabelece a finalidade de um cenário. O cenário deve descrever o modo em que este objetivo deve ser alcançado.

Contexto: Descreve o estado inicial de um cenário, suas pré-condições, o local (físico) e tempo.

Recurso(s): identifica os objetos passivos com os quais lidam os atores.

Ator(es): Detalha as entidades envolvidas nos cenários.

Episódio(s): 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 e exceções.

Esta proposta de cenários tem sido utilizada com sucesso por nosso grupo de pesquisa em experimentos recentes [Breitman98], [Hadad97]. Este modelo de cenários bem como mais detalhes sobre a *Requirements Baseline* podem ser encontrados em [Leite97].

3. Evolução dos cenários

É fato que sistemas evoluem durante seu ciclo de vida e, desta forma, os requisitos não devem ser tratados como um contrato [Lehman91]. Esta preocupação tem permeado os trabalhos que nosso grupo de pesquisa vem desenvolvendo na área de requisitos. Tanto em estudos de caso [Breitman98] [Hadad97] quanto em tutoriais de utilização de cenários temos dado enfoque ao aspecto evolutivo dos cenários. Nesta seção apresentaremos um resumo dos nossos resultados.

Através da observação de estudos de caso fomos capazes de compreender melhor o modo em que os cenários evoluem. Ficou claro que existem dois tipos de mudanças, aquelas que envolvem um conjunto de cenários e outras que se dão dentro do âmbito do mesmo cenário. Chamamos a primeira de mudanças *inter cenário* e a segunda de *intra cenário*, respectivamente. Na figura 2 ilustramos a evolução de um cenário durante vários estágios de seu desenvolvimento. Dois aspectos fundamentais da evolução estão ilustrados nesta figura, a modificação do conteúdo dos cenários e seu intrincado relacionamento com o restante da documentação do sistema. Mais detalhes podem ser encontrados em [Breitman98].

A medida em se desenrola o sistema notamos que cada um dos cenários passava por um processo de refinamentos sucessivos, no qual seu conteúdo sofria modificações. Também observamos que os cenários evoluíam em grupo. Relações estáticas entre cenários apontavam para a existência de forte acoplamento entre os cenários, onde pudemos reconhecer relações de equivalência, coincidência parcial e total, entre outras. A partir do conjunto de relações identificadas foi possível identificar um conjunto de operações que podem ser realizadas sobre cenários. Uma taxonomia para relações e operações está resumida na tabela I a seguir.

Evolução de Cenários		
Inter cenário		Intra cenário
Relações	Operações	Operações
Coincidência parcial	Fundir	Divisão
Contenção	Encapsular	Modificação
Equivalência	Consolidar	Exclusão
		Entrada

Tabela I – Taxonomia para evolução inter e intra cenários

Generalizamos as relações estáticas em três categorias: coincidência parcial, equivalência e contenção. **Coincidência parcial** é definida quando dois (ou mais cenários) tem pelo menos um episódio em comum ou na eventualidade de dividir os mesmos recursos e ter pelo menos um ator em comum. **Contenção** por outro lado é definida como um subconjunto, onde um cenário tem de conter todos os episódios contidos no outro. Para estabelecer uma relação de contenção também é necessário que que ambos os cenários tenham atores em comum e estejam situados em um mesmo contexto. Finalmente a relação de equivalência é definida quando dois cenários possuem episódios idênticos e envolvem os mesmos atores porém lidam com restrições e/ou exceções diferentes.

Uma vez identificada a existência de uma das relações mencionadas entre um par de cenários, podemos realizar operações sobre os mesmos. Foram identificadas três operações do tipo inter cenário e quatro do tipo intra cenário. A operação fundir se caracteriza quando observada a existência de forte coincidência parcial entre cenários. A operação de encapsulamento, por sua vez é disparada quando é observada uma coincidência parcial fraca entre os cenários. Fazemos distinção entre coincidência parcial fraca e forte de modo a distinguir qual operação utilizar. As heurísticas de classificação de intensidade de coincidência podem ser encontradas em [Leite97-b]. Finalmente a operação de consolidação consiste na união de dois cenários equivalentes. É preciso notar que até o presente momento somente identificamos operações entre pares de cenários. Estamos investigando a existência de operações que envolvam um número maior de cenários.

Reconhecemos quatro relações intra cenários. A primeira delas, divisão consiste na separação de um cenários em dois ou mais cenários independentes. Observamos que esta operação surge como um resultado de duas situações. A primeira possibilidade é o refinamento da informação ao longo do desenvolvimento. Uma vez que se atinge um

estágio de maior compreensão do macrosistema, notamos uma tendência a separar grandes blocos de eventos em cenários mais enxutos facilitando a gerência. A segunda possibilidade é uma imposição da parte de projeto, que força uma divisão artificial dos cenários. Em qualquer um dos casos a operação de divisão é um expediente válido para a redução da complexidade. As operações restantes consistem nas atividades clássicas de gerência de dados. Mais detalhes sobre esta taxonomia, bem como exemplos explicativos podem ser encontrados em [Breitman98].

4. Suporte automatizado

O desenvolvimento de um sistema implica na compreensão do macrosistema em que este será futuramente inserido. Para tal é necessário modelar uma série de aspectos que levem em conta os atores, entidades e ações envolvidas. O macrosistema como um todo não é uma entidade estática, se modifica ao longo do desenvolvimento. A especificação do sistema deve ser capaz de refletir estas mudanças. Na seção anterior apresentamos uma discussão dos aspectos evolutivos dos cenários de uma aplicação. O exemplo ilustra uma série de transformações que os cenários sofrem ao longo de seu desenvolvimento. Vários aspectos, tais como a manutenção da consistência entre diferentes versões, estão envolvidos tornando mais complexa a rastreabilidade da informação. Nesta seção apresentaremos uma discussão sobre a possibilidade de fornecer apoio automatizado, mesmo que parcial, à evolução de cenários.



Figura 2 – Evolução de cenários

Nossa experiência no desenvolvimento de sistemas utilizando cenários apontou diversas áreas que poderiam se beneficiar de apoio automatizado. Uma característica básica do enfoque de cenários é a geração de um volume extenso de informação, o que dificulta a realização de consultas específicas e a manutenção da consistência entre cenários ao longo do processo evolutivo. Outro problema é a visualização da dinâmica dos cenários. A partir de um único cenário vários resultados são possíveis, levando-se em conta o contexto em que este se encontra e suas possíveis exceções. Notamos a necessidade de apoio à simulação de cenários. Por último, a própria estrutura dos cenários sugere uma leitura não-linear, recheada de referências a outros documentos, o que os torna candidatos naturais para uma implementação no formato hipertextual. Estas observações estão resumidas na tabela II, a seguir.

Facilidades Hiperímia	Facilidades de Banco de Dados	Gerência de Configuraça
<ul style="list-style-type: none">• Permitir leitura não-sequencial de documentos• Permitir a incorporação de outros documentos• Simulações• Mapas	<ul style="list-style-type: none">• Mecanismos de visão:• Ponto de vista de ator específico,• Exceções• Atualizações e pesquisas na base de dados• Listagens	<ul style="list-style-type: none">• Manutenção da consistência• Manutenção da consistência• Registro de tomada de decis

Tabela II – Requisitos para a apoio automatizado à gerência da evolução de cenários [Breitman98]

4.1 Facilidades Hiperímia

Da maneira com que concebemos os cenários, i.e., utilizando vocabulário referenciado em outra documentação e recheados de referências a outros cenários, estes são candidatos naturais a representação através de hipertexto. Hipertexto é definido como a tecnologia de leitura e escrita não-sequencial [Berk91]. Hiperímia, por sua vez, incorpora facilidades de áudio, vídeo e gráficos ao hipertexto. Acreditamos que aliar a tecnologia de hiperímia à de cenários pode ser benéfico.

Por outro lado, disponibilizar uma série de opções em cada cenário pode vir a confundir usuários. Esta é uma dificuldade inerente a sistemas hiperímia e existem várias propostas que permitem evitar esta situação conhecida como "*perdido no hiperspaço*". Entre as possibilidades se encontram mapas de navegação e a criação de marcadores ou "*migalhas de pão*" (breadcrumbs) que servem como indicadores de lugares já visitados [Shneiderman91]. No caso de adaptar estas técnicas ao nosso caso específico, acreditamos que não somente darão conta de evitar que o usuário se perca durante a leitura, mas também poderão trazer benefícios na utilização de cenários

que não tinham sido, até agora, levados em conta pela comunidade de requisitos.

4.2 Facilidades de Banco de Dados

Cenários basicamente descrevem situações do mundo real que, com frequência, podem ter mais de uma consequência. Deste contexto surge uma questão de ordem prática, qual seria a maneira mais adequada de se organizar um conjunto de cenários de modo a facilitar sua validação junto usuários? Devemos levar em conta que cada cenário pode admitir várias exceções que por sua vez conduzem a outros cenários. Este fato impede que os cenários sejam organizados sequencialmente. Outra dificuldade é garantir que todas as possibilidades foram exploradas. Conforme colocado anteriormente cada cenário pode se subdividir em uma série de ramificações conduzindo a novos cenários e assim sucessivamente. Chamamos este fato de efeito cascata. O efeito cascata dificulta grandemente a leitura dos cenários pois leva a uma explosão de possibilidades em curto espaço de tempo.

Uma forma de combater o efeito cascata na leitura de cenários seria adotar mecanismos de visão parcial similares aos utilizados por bancos de dados. Para adotar este enfoque é necessário contemplar o conjunto cenários (e documentação associada) como elementos de um grande repositório de dados. Este repositório tipicamente conteria os elementos nucleares dos cenários, i.e., atores, contexto, objetivos, recursos e episódios além de entradas no LAL. Desta forma cada visão da base de dados seria um corte longitudinal que englobaria um subconjunto destes elementos. Um exemplo seria verificar todos os cenários a partir do ponto de vista de um ator específico. Este corte conteria todos os cenários onde este ator estivesse presente. Outra possibilidade é observar a evolução de um mesmo cenário em momentos diferentes durante o desenvolvimento.

A figura 3, a seguir, ilustra estes dois exemplos.

Figura 3 – Repositório de dados de cenários. O primeiro plano ilustra todos os cenários que envolvem um ator em especial, o plano β ilustra um mesmo cenário em diferentes contextos

4.3 Gerência da Configuração

Podemos afirmar que, do ponto de vista do apoio à evolução natural dos cenários, mecanismos de gerência de configuração serão aqueles que trarão maiores benefícios. Chamamos de evolução natural o refinamento da informação básica dos cenários elicitada ao início do processo. Este se dá como resultado do aumento da compreensão do macrosistema e suas implicações. Este tipo de controle pode ser executado através de mecanismos clássicos de controle de versão, que fornecem um histórico dos cenários e um *log* das modificações que estes sofreram ao longo do tempo.

Na seção anterior mostramos os resultados de um estudo de acompanhamento da evolução dos cenários de uma aplicação. A partir destes resultados, concluímos que durante o desenvolvimento de um sistema cenários sofrem grandes modificações que podem ser traduzidas por um conjunto de operações, ilustrado pela tabela I. A partir deste estudo observamos que a evolução de cenários está longe de ser um processo de refinamento e que, portanto, poderia ser gerenciado pelo controle de versões apenas. Na realidade, o processo evolutivo de cenários tem um caráter transformacional. Raros são os cenários que permanecem inalterados durante a evolução do processo; união com outros cenários, separação, consolidação e eliminação são algumas das possibilidades, além do processo de refinamento da informação mencionado acima. Desta forma surge a necessidade de criar mecanismos de controle que permitam a gerência destas operações.

Se por um lado o grande número de operações sofridas pelo conjunto de cenários torna a gerência de seu processo evolutivo mais complexo, também o torna mais rico em informação. Acreditamos que o estudo dos padrões transformacionais em cenários podem vir a trazer contribuições ao entendimento geral do macrosistema auxiliando, não só o desenvolvimento, mas possibilitando previsões futuras. Acreditamos que um mecanismo válido seria o emprego de um sistema para o armazenamento do *rationale* que antecede a realização das operações. Se pudéssemos armazenar as razões que ditaram a realização de uma operação, e.g., união de dois cenários,

talvez fosse possível determinar padrões de modo a automatizar este processo. Outra vantagem da captura do *rationale* de decisões é fornecer apoio a manutenção do sistema. Acredita-se que vários problemas inseridos através de um código que sofreu manutenção é resultado direto de mudanças nos requisitos do sistema realizadas inadvertidamente. É possível que em presença documentação de melhor qualidade este problema seja minimizado.

5. Conclusão

Neste artigo trazemos a tona a problemática envolvida na gerência da evolução de cenários. Mostramos resultados de nossos estudos na utilização de cenários para especificação de sistemas. Estes resultados são traduzidos em um conjunto de operações classificadas em inter e intra cenários. A partir dos últimos, fomos capazes tecer comentários sobre o apoio automatizado à função de gerência do processo evolutivo de cenários. Três áreas foram identificadas que podem vir a contribuir no suporte ao desenvolvimento de sistemas baseados em cenários. Estas são as áreas de hipermídia, banco de dados e gerência da configuração. A idéia básica é fazer um aproveitamento de avanços da pesquisa e prática destas áreas e adaptá-las no apoio automatizado à utilização de cenários.

Pretendemos traduzir a discussão proposta neste artigo em um conjunto de requisitos para a construção de uma ferramenta de apoio automatizado à gerência da evolução de cenários. Esta ferramenta deve levar em conta as dificuldades apontadas anteriormente, além de tirar proveito de vantagens possíveis somente em implementações utilizando o meio computacional. Um exemplo é a incorporação de facilidades hipermídia, que permitem uma leitura não-sequencial da informação, e a incorporação de outros documentos. Outras possibilidades são a criação de novos mecanismos de visão dos dados e a gerência da configuração. Esta ferramenta também deve ser compatível com ferramentas de criação de cenários, tais como a proposta em [Antonelli98]. Acreditamos que tal ferramenta será útil durante o desenvolvimento de sistemas baseados em cenários.

Referências Bibliográficas

- [Antonelli98] Antonelli, L. – ReCase – trabalho final de graduação – UNLP – Faculdade de Ciências Exatas, Departamento de Informática – Argentina, 1998.
- [Berk91] Berk, E. - Hypertext and Hypermedia Handbook – Emily Berk and Joseph Devlin editors, McGraw Hill Software Engineering Series – New York, 1991.
- [Breitman97] Breitman, K.; Leite, J.C.S.P. - Scenario Evolution: observations from a case study - Proceedings of the International Conference on Requirements Engineering, IEEE Computer Society Press, pp. 214-221, 1998.
- [Hadad97] Hadad, G., Kaplan, G., Oliveros, A., Leite, J.C.S.P - Construcción de Escenarios a partir del Léxico Extendido del Lenguaje – SoST’97 (Simposio de Teconologia de Software) 26 Jornadas Argentinas de Informática e Investigacion Operativa, Buenos Aires, Agosto,1997.
- [Lehman91] Lehman, M. - Software Engineering the Software Process and their support, IEEE Software Engineering Journal, IEEE Computer Society Press, vol.6 pp 243-258,1991.
- [Leite93] Leite, J.C.S.P.; Franco, A.P.M.Franco - A Strategy for Conceptual Model Acquisiton. Proceedings of the IEEE International Symposium on Requirements Engineering, IEEE Computer Society Press, Pags. 243-246, San Diego 1993.
- [Leite97] Leite, J.C.S.P.; Rossi, G.; Maiorana, V.; Balaguer, F.; Kaplan, G.; Hadad, G.; Oliveros, A. - Enhancing a Requirements Baseline with Scenarios. Requirements Engineering Journal: Springer-Verlag London Limited Vol. 2, N. 4, Pags. 184 -- 198, 1997.
- [Leite97-b] Leite, J.C.S.P.; Breitman, K.K. - Scenario Evolution: observations from a case study – relatório técnico - Departamento de Informática PUC-Rio – Dezembro, 1997.

[Maiden98] Maiden, N.A.M.; Minocha, S.; Manning, K.; Ryan, M. – CREWS-SAVRE: Systematic Scenario Generation and Use- Proceedings of the International Conference on Requirements Engineering, IEEE Computer Society Press, pp.148-155, 1998.

[Shneiderman91] Shneiderman, B.; Kreitzberg, C.; Berk, E. – *Editing to Structure a Reader's Experience* Hypertext and Hypermedia Handbook – Emily Berk and Joseph Devlin editors, McGraw Hill Software Engineering Series – New York, 1991.

[Zorman 95] Zorman, L. - Requirements Envisaging by Utilizing Scenarios (Rebus) - Ph.D. Dissertation, University of Southern California, 1995.