Mapeando Cenário & LEL para o modelo Statechart: Uma Estratégia para Verificação da Especificação de Requisitos

Silvia Pereira de Azevedo Sousa [1], Carla Silva [1]

1 Centro de Informática/UFPE
Av. Jornalista Aníbal Fernandes 50740-560, Recife/PE, Brazil
+55 81 2126-8430
{spas, ctlls}@cin.ufpe.br

Abstract. A fase de especificação de requisitos de software (SRS) é fundamental no processo de desenvolvimento de software. O uso de cenários é bastante popular para especificar requisitos e, normalmente, são descritos em linguagem natural (LN), dando margem a requisitos ambíguos, incompletos e inconsistentes. Esses problemas são preocupantes, pois as especificações em LN podem prejudicar a qualidade final do software. Nesse sentido, uma técnica baseada em Cenário & LEL (Language Extended Lexicon) - C&L - pode ser a solução adequada para reduzir esses problemas, pois é uma técnica de cenários controlada pelo LEL, que por sua vez é um glossário que define um conjunto de palavras que pertencem a um domínio. Apesar de C&L ser uma LN controlada, ela não garante a completude e consistência dos requisitos. Surge a necessidade de verificar essas especificações por meio de um mapeamento para uma linguagem mais formal. O presente trabalho propõe um mapeamento de C&L para o modelo de statechart com o objetivo de verificar a completude e consistência dos requisitos especificados nos cenários.

Keywords: Especificação de Requisitos de Software (SRS), Cenários, LEL, mapeamento, Statecharts.

1 Introdução

Na Engenharia de Requisitos (ER) existem técnicas que auxiliam o engenheiro de requisitos a compreender o universo do problema como, por exemplo, as técnicas baseada em cenários (casos de uso, *storyboard*, C&L, etc) que têm sido muito utilizadas para entender, modelar e validar os requisitos de usuários.

Para facilitar o processo de comunicação entre as partes interessadas, pode-se fazer uso de uma linguagem natural (LN). Segundo [8], a LN é amplamente utilizada na descrição de software, já que proporciona um canal de comunicação entre os intervenientes no processo de construção de software. Além disso, a LN é facilmente associada com métodos e linguagens para especificação de requisitos, tais como os cenários apresentados em [2] e o uso de léxicos apresentado em [3].

Segundo [9], cenário é uma descrição parcial do comportamento da aplicação que ocorre em um dado momento em um contexto geográfico específico - uma situação. Já o LEL segundo [3], é um glossário que define um conjunto de palavras que pertencem a um domínio. Ele está ligado a uma ideia simples: *"understand the language of the problem, without worrying about understanding the problem*" e o seu principal objetivo é capturar o vocabulário da aplicação e sua semântica, adiando a compreensão da funcionalidade da aplicação [10].

Para [4], a aborgem C&L pode ser utilizada na fase da elicitação de requisitos como também ao longo de todo o processo de desenvolvimento do software permitindo a validação com o cliente/usuário. Essa abordagem também contribue para o desenvolvimento de cenários mais controlados, efetivos e não ambíguos, diferenciando-

se da abordagem de casos de uso por considerar uma visão mais abrangente, incluindo a descrição de cenários e informações sobre o contexto no qual o sistema será inserido.

No entanto, especificações de requisitos baseadas em LN, como o C&L, necessitam serem verificadas ainda na fase inicial da produção do software, visando melhorar a qualidade do produto. Os autores [6] afirmam que mapear cenários para técnica de statecharts ajuda a encontrar muitas omissões, ambiguidades e inconsistências nas especificações.

Segundo [1], statecharts são formalismos visuais que podem ser usados para especificar comportamentos do sistema garantindo que as especificações sejam mais corretas, além de permitir gerar diagramas compactos e bastante expressivos, ou seja, corroborando para especificações mais gerenciáveis e compreensíveis.

Motivado por este contexto, o trabalho propõe um mapeamento de C&L para statechart, com o intuito de promover uma verificação nos cenários, visando identificar problemas de incompletude e inconsistência mais cedo, corroborando para uma especificação de requisitos com melhor qualidade. Assim, espera-se identificar uma quantidade maior de problemas nos requisitos já nessa fase.

O trabalho está organizado da seguinte maneira. Na Seção 2, será apresentado os trabalhos relacionados. Na Seção 3, será demonstrado o mapeamento de C&L para statecharts. Na seção 4, será apresentado um exemplo ilustrativo. A Seção 5, expõe as conclusões e os direcionamentos para trabalhos futuros.

2 Trabalhos Relacionados

Algumas pesquisas têm mostrado a importância de verificar cenários descritos em LN, através de uma linguagem mais formal, como os trabalhos [7] e [6].

Em [8], os autores descrevem os requisitos iniciais como cenários e estes cenários são transformados em diagramas de atividades de maneira automatizada através do uso da ferramenta C&L para apoiar testes automatizados. As trajetórias do diagrama de atividades são geradas a partir das interações entre os episódios, exceções e restrições e são utilizados para a geração de cenários de teste iniciais através do uso de um gráfico de pesquisa e estratégia de combinação de trajetória. Uma limitação deste trabalho é em relação a geração de teste de exceções e requisitos não funcionais e também esta abordagem não é aplicável a situações que envolvem simultaneidade. O diferencial da nossa abordagem é que podemos usar as sentenças não sequencias do cenário para representar situações de simultaneidade usando o Estado AND, podemos também capturar os NFRs através de Annotations e representá-los no modelo statecharts.

Outro trabalho é o descrito em [7], que apresenta uma abordagem baseada em Petri-net com o objetivo de analisar de maneira eficaz os cenários que descrevem os requisitos de entrada de um sistema. Para permitir uma análise automatizada, os cenários e suas interações são traduzidas em conformidade com o metamodelo Petri-net através de um procedimento sistemático (mapeamento de cenário para elementos Petri-Net), além de utilizar um algoritmo para transformar de cenário para Petri-Net e um algoritmo de integração de Petri-nets. A nossa abordagem diferencia-se desta, pois define regras de mapeamento para transformar a abordagem C&L em modelo statechart.

Uma abordagem bastante correlata com a nossa pesquisa, é o trabalho de [6] que faz uso de uma abordagem baseada em cenários que valida e testa o sistema através do uso de statechart. Essa validação de cenário é acompanhada por medidas de verificação. A verificação é apoiada por um método (SCENT-Method) de formalização. Uma limitação que este trabalho apresenta é em relação ao processo de formalização, ou seja, o mapeamento de ações em cenários de linguagem natural para estados ou transições não é bem definida e clara. Diferentemente deste trabalho, a nossa abordagem faz uso de regras de mapeamento que permitem transformar o C&L em statecharts, permitindo que os modelos de statechart obtidos por parte de desenvolvedores distintos sejam muito semelhantes ou iguais.

3 Mapeando C&L para Statecharts

Com o intuito de mostrar a conformidade existente entre os elementos de C&L e Statecharts, foi elaborada uma tabela que define regras de mapeamento (ver Tabela.1) para derivar Statecharts a partir de C&L, no qual o objetivo é verificar a qualidade das especificações através de uma linguagem semi- formal.

Neste mapeamento, os atores não são considerados, pois devem aparecer ao menos um ator em cada apisódio, além do que o mapeamento considera os episódios que são ações e que dão uma descrição operacional do comportamento do sistema.

Tabela 1. Regras de mapeamento

	Elementos C&L	**Elementos Statecharts**
1	Título	Super-Estado
2	Episódio sentença Simples	Estado (Tarefas sequenciais)
3	Resultado da execução da sentença	Evento da Transição
4	Episódio Sentença condicional	Estado XOR Transição de um episódio de sentença simples para outro episódio de sentença excepcional.
5	Episódio Sentença *Loop*	*Estado*
		Transição com repetições no mesmo estado
6	Episódio Sentença Opcional	Evento de transição (tarefa opcional pode ser executada repetidamente ou possivelmente pode não ser executada).
7	Episódio Sentença Concorrente	Estado ortogonal (Estado AND) – vários estados executando paralelamente.
8	Contexto: Pré e Pós-condição; Pré e Pós-condição; Restrição	*Annotations* (Informações adicionais como dados: entrada, saída, intervalos esperados; desempenho e qualidades).

A tabela 1 está organizada contemplando os elementos de C&L e do statechart respectivamente. Na sequência, vamos apresentar as heurísticas para apoiar a transformação explicando como os conceitos de C&L podem ser acomodados dentro do Statecharts, a fim de representar o comportamento do sistema, a citar:

Em C&L o título identifica o cenário utilizado e deve ser exclusivo. Em statecharts, o super-estado representa o estado maior delimitando a aplicação do sistema. Portanto, na linha (1) da tabela o título é mapeado como super-estado em statecharts (ver Tabela 1).

O episódio de sentença simples em C&L representa o fluxo principal necessário para completar o cenário, representado por uma sequência de sentenças onde tudo funciona como esperado. Cada sentença é um episódio que pode ser visto como uma tarefa. No statechart, um estado é representado pela execução de tarefa. A realização de uma tarefa (estado) conduz a uma transição para outro estado que representa outra tarefa a ser executada na sequência. Assim, uma sentença do episódio simples é mapeada como uma estado em statechart (linha (2) da tabela).

O resultado da sentença em C&L representa o resultado após a execução da tarefa representada na sentença. Já no statechart, a conclusão de um estado causa uma transição para outro estado. Note que a transição utiliza eventos e/ou condições para que um estado mude para o outro. O evento é o resultado da conclusão do estado e a condição é algo que deve ser verdadeiro, ou então a transição não será executada. Sendo assim, o resultado de uma sentença é mapeado como um evento de uma transição no statechart (linha (3) da tabela).

O episódio de sentença condicional em C&L é representado por um fluxo de exceção que apresenta sentenças alternativas que dependem ser de uma condição interna ou externa para serem executadas. No statechart, o estado XOR representa um fluxo alternativo permitindo estados alternativos. Assim, a transição parte de um estado para

estados alternativos diferentes a depender da condição satisfeita. Desse modo, uma sentença condicional é mapeada como um estado destino da transição XOR no statechart (linha (4) da tabela).

O episódio de sentença loop em C&L identifica repetições na estrutura da sentença até que a condição seja satisfeita. Em statecharts, um loop é representado como uma transição que se repete em um mesmo estado até que uma condição seja satisfeita. Desta maneira, a sentença loop é mapeada como um estado que possui uma transição apresentando uma condição para que o evento seja modificado e a tarefa (estado) seja concluída no statechart (linha (5) da tabela).

O episódio de sentença opcional em C&L identifica ações que podem ou não serem executadas. Em statechart, uma execução opcional dentro de uma sequência, possivelmente uma das tarefas (estados) podem não ser executadas sempre, podem ser executadas zero ou mais vezes ou podem ser executados uma ou mais vezes. Assim, o episódio de sentença opcional é mapeado como uma tarefa opcional no statechart (linha (6) da tabela).

Sentenças concorrentes em C&L ocorrem paralelamente e são limitadas pelo símbolo "#". No statechart, o estado ortogonal (Estado AND), representa vários estados executando ao mesmo tempo. Desta forma, sentenças concorrentes são mapeadas como estados ortogonais/paralelos em statechart (linha (7) da tabela).

Em C&L as pré e pós-condições são descritas como sentenças declarativas envolvendo membros relevantes como (ator/recurso/sistema) adicionando informações de qualidade. Diferentemente, as pré e pós-condições do contexto são usados em situações do sistema antes do cenário ser iniciado e quando os episódios são realizados respectivamente. Esses atributos são importantes na modelagem de sistemas concorrentes [5]. Já as restrições são sentenças declarativas que restrigem a qualidade do sistema. No statechart, as informações importantes como dados, desempenho e qualidades são capturados em Annotations. Desta maneira, na (linha (8) da tabela) as pré, pós- condições e as restrições são mapeadas como Annotations no statechart.

4 Exemplo Ilustrativo

Os modelos de origem apresentados nas Figuras (1 e 2) foram retirados do trabalho de [7]. No cenário "Submit order" são identificados 15 eventos (13 fluxos principais – episódios e 2 fluxos excepcionais). Neste cenário são descritos interações entre o Sistema Broker Online e os serviços de parceria com fornecedores (SupplierA, SupplierB e SupplierC) representados pelos episódios 10, 11 e 12 usando a linguagem do cenário. O cenário "Submit order" interage com os episódios 10, 11 e 12 que são executados paralelamente em uma ordem não sequencial limitada pelo símbolo "#" como, por exemplo, o episódio 10 que foi detalhado na Fig. 2 (os episódios 11 e 12 faz uso do mesmo cenário).

TITLE: ***Submit Order***
GOAL: Allow customers to find the best supplier for a given order.
CONTEXT:
PRE-CONDITION: The Broker System is online AND the Broker System welcome page is being displayed
ACTOR: Customer, Broker System
RESOURCES: Login page, Login information, Order
EPISODES
1. The Customer loads the login page
2. The Broker System asks for the Customer login information
3. The Customer enters her login information
4. The Broker System checks the provided login information
5. The Broker System displays an order page
6. The Customer creates a new Order
7. **DO** the Customer adds an item to the Order **WHILE** the Customer has more items to add to the order
8. The Customer submits the Order
9. The Broker System broadcast the Order to the Suppliers. **Post-condition**: An Order has been broadcasted
10. **#** SUPPLIERA BID FOR ORDER
11. SUPPLIERB BID FOR ORDER
12. SUPPLIERC BID FOR ORDER **#**
13. PROCESS BIDS.

EXCEPTIONS
4.1 **IF** the Customer login information is not accurate **THEN** the Broker System displays an alert message
8.1 **IF** the order is empty **THEN** the Broker System displays an error message

Fig. 1. Modelo C&L *Submit Order*.

TITLE: ***SupplierA bid for order***
GOAL: SupplierA has bidded
CONTEXT: Create a Bid for an Order
PRE-CONDITION: An Order has been broadcasted
POST-CONDITION: SupplierA has bidded
ACTOR: SupplierA, Broker System
RESOURCES: Order, Bid
EPISODES
1. SupplierA receives the Order and examines it
2. SupplierA submits a Bid for the Order
3. The Broker System sends the Bid to the Customer
EXCEPTIONS
1.1 **IF** SupplierA can not satisfy the Order **THEN** SupplierA passes on the Order

Fig. 2. *SupplierA bid for order Scenario*

O modelo statechart completo é gerado a partir do cenário "*Submit order*" (modelo de origem) apresentado na figura 1. Devido ao espaço não iremos representar o modelo completo de todas as sentenças do cenário. Para exemplificar como um elemento C&L é derivado para statechart selecionamos uma parte da trajetório no modelo (Ver figura 3). A trajetória selecionada, representa a sentença simples, pois percorre os estados E2, E3 e E4 de maneira sequencial onde tudo funciona como esperado. A partir do estado E4, a trajetória apresentada corresponde a sentença condicional que representada por episódios de sentença excepcional (E4.1) e (E4.2) respectivamente. Se ocorrer falha em (E4.1) e a condição (c1) não for atendida, o sistema dispara o evento (t5) e retorna para o estado (E2) e segue todo o fluxo novamente, caso contrário se a condição c2 for atendidada o estado E4.2 é executado e o sistema dispara o evento (t6) mudando para o estado (E5).

Note que, a partir das trajetórias modeladas no statechart pode-se detectar, por exemplo, inconsistências a medida que o modelo estiver sendo construído ao aplicar as regras de mapeamento.

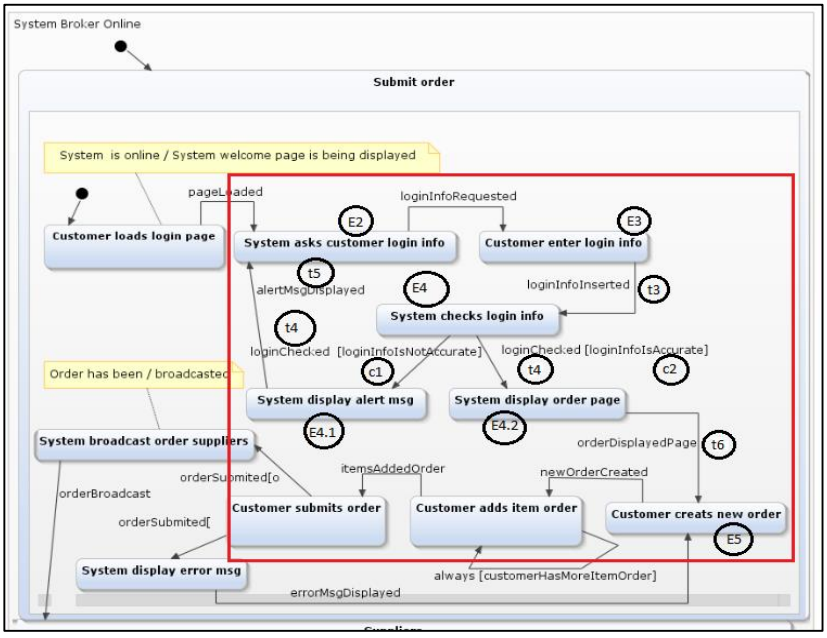

Fig.3. Trajetória da sentença simples e condicional no statechart.

5 Conclusão e Trabalhos Futuros

Após a realização da modelagem, notou-se que esta representação ajuda a encontrar alguns erros de inconsistência durante o processo de transformação, pois é possível detectar algumas omissões e erros ao aplicar as heuristicas de mapeamento. Em contra partida, a abordagem não verifica o cenário por completo, muitos problemas ainda podem permanecer escondidos.

Como trabalhos futuros têm-se: Uma vez que o modelo de origem é mais formal que o modelo de destino, a partir do elemento *"Annotations"* presente no statechart será criado casos de teste ainda na fase inícial do desenvolvimento. Esses casos de testes podem ser benéficos à qualidade dos artefatos a serem produzidos e posteriormente colaborar para automatizar a fase de teste do desenvolvimento de software.

Agradecimentos. Agradecemos ao grupo de pesquisa LER (*Requirements Engineering Laboratory*) da Universidade Federal de Pernambuco e a FACEPE (Fundação de Amparo à Ciência e Tecnologia do Estado de Pernambuco) por apoiar esta pesquisa.

References

1. Harel, D.: Statecharts: A visual formalism for complex systems. In.: Science of Computer Programming, 8, (3), 231-274 (1987).
2. Leite, J. C. S. P. et al.: A Scenario Construction Process. In.: RE. v 05, pp. 38 – 61 (2000).
3. Leite, J. C. S. do P.; Franco, A. P. M.: A Strategy for Conceptual Model Acquisition. In.:IEEE Xplore Digital Library, (1993).
4. Leite, J.C.S.P.; Rossi, G.; Balaguer; F. Maiorana, V.: Enhancing a requirements baseline with scenarios. In Proceedings of the Third IEEE International Symposium on Requirements Engineering – RE'97, IEEE Computer Society Press, pp. 44-53 (1997).
5.Leite, J. C. S. P.; Franco A. P. M.:O Uso de Hipertexto na Elicitação Linguagens da Aplicacão. In: Anais de IV Simpósio Brasileiro de Engenharia de Software. SBC, Brasil, pp 134–149. 13 (1990).
6. Ryser, J.; Glinz, M.: A Scenario-Based Approach to Validating and Testing Software Systems Using Statecharts. In.: 12th International Conference on Software and Systems Engineering and their Applications, ICSSEA. Proceedings: CNAM, Paris, France (1999).
7. Sarmiento, E. et al.: Analysis of Scenarios with Petri-Net Models. In.: SBES, Belo Horizonte – MG (2015).
8. Sarmiento, E. et al.: C&L: Generating Model Based Test Cases from Natural Language Requirements Descriptions. In.: RET, Karlkrona, Sweden (2014).
9.Zorman, L.: Requirements envisaging by utilizing scenarios (Rebus). PhD dissertation, University of Southern California, 11 (1995).
10. Vilela, J.F.F.: On the behavior of context-sensitive systems. In.: CibSE - WER (Workshop em Engenharia de Requisitos), Lima – Peru (2015.