ChAPTER: Um Método para Geração de Cenários de Testes para Linhas de Produto de Software Sensíveis ao Contexto

Ismayle de Sousa Santos[1,a], Rossana Maria de Castro Andrade[1,b], Pedro de Alcântara dos Santos Neto[2]

[1]Mestrado e Doutorado em Ciência da Computação (MDCC) – Universidade Federal do Ceará (UFC) – Fortaleza, Ceará, Brasil

[2]Departamento de Computação – Universidade Federal do Piauí (UFPI) – Teresina, Piauí, Brasil

{ismaylesantos,rossana}@great.ufc.br; pasn@ufpi.edu.br

Abstract. *The Software Product Line (SPL) paradigm has emerged as a means to achieve large-scale reuse. When this paradigm is used to develop context aware applications, we have a Context Aware SPL. Testing this type of line is more complex compared to a traditional SPL. In this scenario, this paper presents a method of generating test scenarios for a Context Aware SPL that uses textual specification of use cases. Are also described in this paper the results of a preliminary assessment conducted with a controlled experiment.*

Resumo. *O paradigma de Linha de Produto de Software (LPS) surgiu como um meio para alcançar o reuso em larga escala. Quando este paradigma é utilizado para o desenvolvimento de aplicações que alteram dinamicamente o seu comportamento ou proveem serviços com base em informações de contexto, tem-se uma LPS Sensível ao Contexto (LPSSC). Por lidar com informações de contexto, testar esse tipo de linha é mais complexo se comparado com uma LPS tradicional. Neste cenário, este artigo apresenta um método de geração de cenários de testes para uma LPSSC que utiliza especificações textuais de casos de uso. Também são descritos neste artigo os resultados de uma avaliação preliminar conduzida na forma de um experimento controlado.*

1. Introdução

A partir do reuso de software é possível obter benefícios como o aumento da produtividade e a redução dos custos de desenvolvimento [Almeida *et al.*, 2007]. Neste cenário, uma Linha de Produto de Software (LPS) pode ser usada para elevar o grau de reuso e maximizar esses benefícios durante o desenvolvimento de várias aplicações. Isso porque com uma LPS, sistemas de software que compartilham características comuns, são desenvolvidos a partir de artefatos comuns de forma sistemática [Northrop 2002].

[a] Bolsista de Doutorado da CAPES
[b] Bolsista do CNPq de Produtividade DT-2 com o número de processo 314021/2009-4

Uma Linha de Produto de Software pode ser chamada de Sensível ao Contexto (LPSSC) [Ferndandes et al. 2011], quando ela for concebida para o desenvolvimento de aplicações sensíveis ao contexto. Este tipo de aplicação adapta o seu comportamento com base em informações de contexto [Wang et al. 2007].

No caso de uma LPSSC é importante destacar que as peculiaridades de aplicações sensíveis ao contexto, tal qual a volatilidade do contexto, trazem uma complexidade adicional à atividade de testes. Dessa forma, testar uma LPSSC envolve lidar ao mesmo tempo com os desafios inerentes aos testes de uma LPS [Ali e Moawad, 2010], que precisam, lidar com os pontos de variação da linha, e aos testes de aplicações sensíveis ao contexto, que devem levar em consideração informações de contexto.

Na literatura é possível encontrar alguns trabalhos que apoiam a geração de teste a partir de casos de uso para uma LPS [Bertolino e Gnesi, 2003; Ali e Moawad, 2010; Lima Neto et al., 2012]. Em geral, essas abordagens analisam os cenários de caso de uso e produzem algum modelo ou máquina de estados para guiar a geração dos testes. Essas abordagens de geração de testes a partir de casos de uso são interessantes por possibilitarem a geração dos testes desde a etapa de requisitos, aumentando assim as chances da identificação precoce de defeitos. Contudo, não foi encontrado nenhum trabalho relacionado à geração de testes para uma LPSSC. Logo, diante dos desafios relacionados ao teste de uma LPSSC, percebe-se a necessidade de métodos e ferramentas de apoio ao teste desse tipo de linha.

Com base no que foi discutido anteriormente, este artigo aborda o problema de geração de testes para linhas de produto de *software* sensíveis ao contexto, propondo um método, denominado de ChAPTER, para a geração de cenários de testes a partir de casos de uso que definem as funcionalidades, variabilidades e informações de contexto. Escolheu-se casos de uso como entrada do método proposto, porque são simples, podem ser criados desde o início de desenvolvimento da LPSSC e não exigem conhecimento específico em alguma linguagem ou ferramenta.

O restante deste artigo está organizado da seguinte forma: na Seção 2 são apresentados os trabalhos relacionados; o funcionamento do ChAPTER é detalhado na Seção 3; a avaliação preliminar do método é apresentada na Seção 4; e, por fim, as considerações finais e trabalhos futuros são descritos na Seção 5.

2. Trabalhos Relacionados

Segundo a revisão sistemática de Lamancha et al. (2009), a maioria das abordagens para geração de testes em uma LPS utilizam casos de uso. Logo, é possível encontrar vários trabalhos relacionados à geração de testes a partir de casos de uso no paradigma de LPS. Contudo, esta seção apresenta apenas os trabalhos relacionados à geração de testes a partir de descrições textuais de casos de uso para uma LPS: [Bertolino e Gnesi, 2003; Ali e Moawad, 2010; Lima Neto et al., 2012; Cai e Zeng 2013].

Bertolino e Gnesi (2003) apresentam o PLUTO (*Product Lines Use case Test Optimization*), que é uma metodologia para derivação de testes a partir de requisitos da LPS descritos como PLUCs (*Product Line Use Case*). Essa metodologia utiliza uma extensão do método de Partição de Categoria [Ostrand e Balcer, 1988], que foi criado originalmente para derivar testes funcionais de especificações em linguagem natural.

Nos artigos de Ali e Moawad (2010) e de Cai e Zeng (2013), os autores propuseram um método baseado em modelos para gerar testes para uma LPS. No caso de Ali e Moawad (2010), os autores propuseram uma técnica que usa o PLUC de Bertolino e Gnesi (2003) e diagramas de atividades para permitir a geração de testes a partir de casos de uso. Cai e Zeng (2013) também usam diagramas de atividades criados a partir de casos de uso para derivar os cenários de testes, mas apresentam um algoritmo para evitar cenários de testes redundantes durante os testes das aplicações da linha.

A ferramenta SPLMT-TE (Lima Neto et al., 2012) gera cenários de testes combinando os cenários dos casos de uso. Com essa ferramenta, os cenários de testes gerados têm os seguintes elementos: identificador, nome, resumo, passos, pré-condição e pós-condição, tipo de variabilidade, referência do caso de uso e caminho do sistema que deve ser testado.

Pelo que foi exposto nesta seção, é possível observar que embora tenham sido encontrados alguns trabalhos relacionados, nenhum deles tratava de uma LPSSC. Dessa forma, este é um diferencial deste trabalho, que procura lidar ao mesmo tempo com as peculiaridades de uma LPS e de aplicações sensíveis ao contexto na geração de testes.

3. ChAPTER

O ChAPTER (*Context Aware software Product line TEsting geneRation method*) gera cenários de testes para LPSSC a partir de descrições textuais de casos de uso. Com relação à proposta deste método, ela foi concebida após uma criteriosa análise das abordagens existentes, onde percebeu-se que uma extensão ou adaptação do método PLUTO de [Bertolino e Gnesi, 2003] para tratar de LPSSC poderia ser interessante, visto que o PLUTO gera cenários de testes diretamente dos casos de uso, reaproveitando os artefatos desenvolvidos em fases anteriores do desenvolvimento.

Dessa forma, o ChAPTER foi proposto tendo como base o método PLUTO [Bertolino e Gnesi, 2003], descrito na Seção 2. Na Tabela 1 é apresentada uma comparação entre o PLUTO e o ChAPTER, destacando a relação deles com os passos da técnica de Partição de Categorias, que foi estendida por ambos os métodos.

Tabela 1. PLUTO x ChAPTER

Passo	Partição de Categorias Original	PLUTO	ChAPTER
01	Análise dos requisitos funcionais para identificar "unidade funcionais" que podem ser testadas separadamente	As unidades funcionais são os PLUCs	As unidades funcionais são os SLICEs e PiECES
02	Para cada unidade funcional, os testadores identificam as categorias relevantes para serem testadas	Normalmente os cenários do caso de uso são uma das categorias	Os cenários e os pontos de variação devem ser categorias
03	Definição das escolhas, que são os valores significantes para cada categoria.	*Tags* são associadas às escolhas para indicar o produto associado	As escolhas podem ser associadas a alguma restrição de contexto
04	Determinar as restrições entre as escolhas de categorias diferentes	Idem	Idem
05	Os cenários de testes são gerados combinando todas as escolhas possíveis para todas as categorias, respeitando as restrições definidas	Idem	Idem

Como apresentado na Tabela 1, além de propor o uso dos cenários como categorias e especificar as restrições de contexto associadas às escolhas, o diferencial do ChAPTER está também no uso de casos de uso que contém informações de variabilidade e de contexto. Esses casos de uso são denominados no ChAPTER de SLICES (*Software product LIne Contextual use casE*), se forem da linha, e PiECES (*Product contextual usE CasEs*) se forem dos produtos da linha.

3.1. Etapas do ChAPTER

Na Figura 1 é apresentado um diagrama de atividades do método proposto, que atua tanto na Engenharia de Domínio quanto na Engenharia de Aplicação de uma LPSSC e gera cenários de testes em três níveis: i) Nível 1: cenários de testes que englobam todas as configurações de todos os produtos da LPSSC; ii) Nível 2: cenários de testes que incluem testes para todas as configurações possíveis de um mesmo produto; iii) Nível 3: cenários de testes específicos de uma dada configuração de um produto da linha.

A primeira etapa inicia com a modelagem dos SLICES. No âmbito deste trabalho, os casos de uso são especificados utilizando o CAPLUC [Santos et al. 2013], que é um *template* de caso de uso, cujo diferencial está em apresentar elementos para acomodar a descrição das variabilidades e informações de contexto de uma LPSSC.

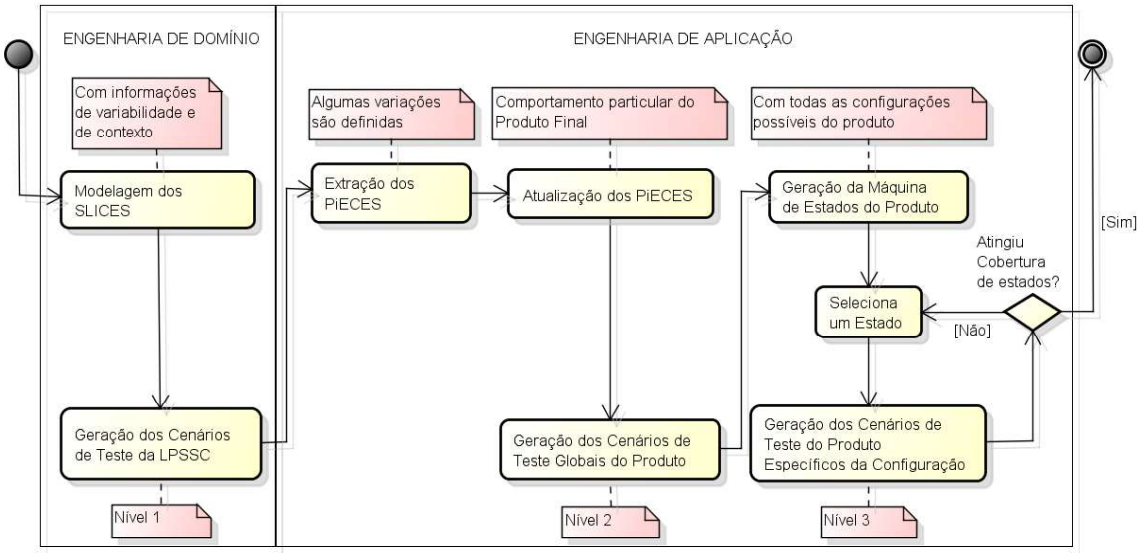

Figura 1. Diagrama de atividades do ChAPTER

Após a especificação dos SLICES, eles podem ser mapeados com as *features* de forma que uma *feature* pode ser um conjunto de SLICES, um único SLICE ou um ponto de variação dentro do SLICE. Ainda na Engenharia de Domínio, a partir dos SLICES da LPSSC são gerados os cenários de teste da linha (Nível 1), conforme uma adaptação do método de Partição de Categorias (ver Tabela 1). Visto que o número de cenários do Nível 1 explode exponencialmente com o número de pontos de variação, o recomendado é utilizar tais cenários para guiar o planejamento dos testes na Engenharia da Aplicação ou para implementar os cenários comuns visando a reutilização dos mesmos durante os testes dos produtos finais.

A segunda etapa do método inicia na Engenharia de Aplicação quando PiECES são gerados a partir dos SLICES por meio da definição dos pontos de variação. Isso

significa, por exemplo, selecionar uma *feature* alternativa dentro das possibilidades de um ponto de variação alternativo. Ressalta-se que ao definir os PiECES o engenheiro da linha estará especificando uma configuração de um produto final.

Em seguida, uma vez que o contexto pode implicar em comportamentos bem específicos para a aplicação final, é previsto no método uma etapa de atualização dos PiECES, para que tais informações sejam especificadas.

Com os PiECES atualizados, é feito um recorte no conjunto de cenários de testes do Nível 1 para extrair os cenários do produto relativos aos PiECES que tiveram variações definidas. Para os PiECES que foram atualizados com informações específicas do produto final, novos cenários de testes são gerados. Como resultado, nessa etapa obtém-se os cenários de testes globais do produto (Nível 2). Nesse ponto, é interessante implementar os cenários de testes comuns, que não variam conforme a configuração da aplicação, para reutilizá-los durante o teste das configurações do produto.

Por fim, a última etapa corresponde à geração de uma máquina de estados para o produto, onde cada estado representa quais *features* estão ativas naquela configuração e a transição entre estados é feita por meio de mudanças de contexto. Dessa forma, essa máquina apresenta todas as configurações possíveis do produto. Em seguida, escolhe-se um desses estados e com base nas *features* ativas é feito um recorte do conjunto de testes globais do produto dando origem aos testes específicos da configuração (Nível 3). Isso significa que cada configuração terá um conjunto de cenários de testes associado. Esse processo de selecionar um estado e realizar um recorte no conjunto de testes do Nível 2 prossegue até se alcançar a cobertura de estados desejada.

É importante destacar que esses testes específicos da configuração são essenciais para verificar se o produto se configura corretamente de acordo com as mudanças de contexto. Isso é possível, pois a máquina de estados possui o registro de qual contexto é necessário para ativar as configurações do produto e, há a especificação do contexto nos casos de uso, torna-se possível identificar quais casos de uso e, por consequência, quais cenários de testes devem estar ativos em uma dada configuração do produto.

Também é importante mencionar que como o método utiliza casos de uso que descrevem o comportamento do sistema (do ponto de vista do usuário), os cenários de teste gerados podem ser utilizados para realização de testes funcionais ou de aceitação. Além disso, o ChAPTER atua tanto na Engenharia de Domínio quanto na Engenharia de Aplicação, o que favorece o reuso dos artefatos de teste, pois os testes gerados na Engenharia de Domínio são refinados na Engenharia de Aplicação. Esse reuso dos artefatos é importante porque possibilita a redução dos custos dos testes de uma LPS.

Para dar suporte à aplicação do ChAPTER foi desenvolvida uma ferramenta denominada de ToChAPTER[3], que apresenta suporte para as três etapas desse método.

3.2. Prova de Conceito

Para uma observação da aplicação do método proposto, o ChAPTER foi usado na geração de cenários de testes para alguns casos de uso da linha Mobiline[4], que engloba o desenvolvimento de guias de visita móveis e sensíveis ao contexto.

[3] Disponível em http://pesquisa.great.ufc.br:8080/tochapter/
[4] http://mobiline.great.ufc.br/

Na Figura 2 é apresentado um excerto do SLICE "Captura Contexto", que especifica como uma aplicação da linha Mobiline pode capturar informações de contexto. Para simplificar este exemplo, optou-se por omitir nesta figura alguns elementos do caso de uso (e.g., nome, descrição), bem como a descrição dos passos. Este caso de uso contém cinco passos, dos quais quatro deles estão associados a alguma variação da linha. Os *indexes* repetidos (com o número 2), indicam os passos associados ao mesmo ponto de variação alternativo, enquanto que, os *indexes* com "*", indicam os passos que possuem restrição de contexto. Mais detalhes sobre o *template* de caso de uso utilizado são encontrados em [Santos et al. 2013].

<table>
<tr><th colspan="2">Passo</th><th>Ações do Usuário</th><th>Ações do Sistema</th></tr>
<tr><td></td><td>1</td><td>...</td><td>...</td></tr>
<tr><td>[Origem]
Alt [Sensor]</td><td>2*</td><td>...</td><td>...</td></tr>
<tr><td>[Origem]
Alt [Memória]</td><td>2</td><td>...</td><td>...</td></tr>
<tr><td>[Origem]
Alt [QR Code]</td><td>2*</td><td>...</td><td>...</td></tr>
<tr><td>[Origem]
Alt [Serviço Externo]</td><td>2</td><td>...</td><td>...</td></tr>
<tr><td colspan="4">Fluxos Alternativos</td></tr>
<tr><td>Fluxo Alternativo 01</td><td colspan="3">...</td></tr>
<tr><td colspan="4">Sumário de Variações Alternativas</td></tr>
<tr><td rowspan="4">[Origem]: "Qual meio utilizado para captura de contexto?"</td><td colspan="2">[Sensor]
(Restrição de Contexto: SENSOR AND SENSORES_AMBIENTE)</td><td>Passo 02</td></tr>
<tr><td colspan="2">[Memória]
(Restrição de Contexto: nenhuma)</td><td>Passo 02</td></tr>
<tr><td colspan="2">[QR Code]
(Restrição de Contexto: NOT_SENSOR OR NOT_SENSORES_AMBIENTE)</td><td>Passo 02</td></tr>
<tr><td colspan="2">[Serviço Externo]
(Restrição de Contexto: nenhuma)</td><td>Passo 02</td></tr>
</table>

Figura 2. Excerto de um SLICE da linha Mobiline

Com o SLICE da Figura 2, gerou-se os cenários de testes (Nível 1) usando a adaptação do método de Partição de Categorias apresentado na Tabela 2. Conforme estabelecido no ChAPTER, os cenários e pontos de variação são colocados como categorias, logo para a primeira categoria temos duas escolhas, o "Cenário Principal" e o "Fluxo Alternativo 01", enquanto que para a segunda categoria existem quatro: "Sensor", "Memória", "*QR Code*" e "Serviço Externo".

Outras categorias também podem ser incluídas para os testes. Nesta prova de conceito, foi considerado que o tipo de informação de contexto era relevante para o teste do SLICE "Captura Contexto". Logo, definiu-se "Informação de Contexto" como uma categoria. Nesse caso, as escolhas dessa categoria foram definidas como "Localização do Visitante" e "Informações do Celular". Como resultado, no total foram definidas três categorias, apresentadas na Tabela 2. A categoria destacada nesta tabela representa a categoria que não foi gerada pelo método, mas sim definida pelo testador.

Observa-se que além da descrição das restrições de contexto, uma restrição foi associada entre as escolhas "Informações do celular" e "Memória", indicando que a primeira só pode ser feita se a segunda tiver sido escolhida. O objetivo de inserir essas restrições é evitar a geração de estados inconsistentes. No caso de uso "Captura de

Contexto", por exemplo, um cenário de teste com "Informações do Celular" e uma variante diferente de "Memória" não é um cenário válido porque no exemplo esse tipo de informações é capturado pela aplicação e fica armazenado na memória.

Tabela 2. Categorias e escolhas para o SLICE "Captura de Contexto"

Categoria	Escolha	Restrição de Contexto
Cenários	Principal	-
	Alternativo 01	-
Origem	Sensor	SENSOR AND SENSORES_AMBIENTE
	Memória	-
	QR Code	NOT_SENSOR OR NOT_SENSORES_AMBIENTE
	Serviço Externo	-
Informação de Contexto	Localização do Visitante	-
	Informação do Celular (SE Memória)	-

Com todas as combinações possíveis das escolhas da Tabela 2, tem-se uma lista de todos os potenciais cenários de testes para todos os produtos da linha com relação ao SLICE "Captura Contexto". Dessa forma, o ChAPTER gera 10 cenários de testes para este caso de uso (ver Tabela 3).

Tabela 3. Cenários de teste da linha para o SLICE "Captura de Contexto"

CT	Contexto	Cenário	Origem	Informação de Contexto
01	SENSOR AND SENSORES_AMBIENTE	Principal	Sensor	Localização do Visitante
02	NOT_SENSOR OR NOT_SENSORES_AMBIENTE	Principal	*QR Code*	Localização do Visitante
03	-	Principal	Memória	Localização do Visitante
04	-	Principal	Serviço Externo	Localização do Visitante
05	-	Principal	Memória	Informação do Celular
06	-	Alternativo 01	Memória	Informação do Celular
07	SENSOR AND SENSORES_AMBIENTE	Alternativo 01	Sensor	Localização do Visitante
08	NOT_SENSOR OR NOT_SENSORES_AMBIENTE	Alternativo 01	*QR Code*	Localização do Visitante
09	-	Alternativo 01	Memória	Localização do Visitante
10	-	Alternativo 01	Serviço Externo	Localização do Visitante

A etapa seguinte é de instanciação dos PiECES a partir dos SLICES. Nesta prova de conceito, foi especificado um PiECE para um Produto A, que apresentava somente as *features* alternativas "Memória", "Sensor" e "*QR Code*". Este PiECE foi definido com base no SLICE "Captura Contexto", retirando-se os trechos relacionados as *features* não selecionadas para o produto, ou seja, a "Serviço Externo". A partir deste PiECE foram então obtidos os cenários de testes globais do Produto A (Nível 2) por meio de um recorte dos cenários de testes da linha. Como o Produto A não tem "Serviço Externo", os cenários de testes ***CT 04*** e ***CT 10*** não são selecionados, ficando assim oito cenários de testes globais do produto.

A última etapa do método inicia com a geração da máquina de estados finita do produto (ver Figura 3), onde cada estado representa um conjunto de *features* ativas e as

transições entre os estados são mudanças de contexto. Com isso, essa máquina de estados apresenta todas as configurações possíveis do produto.

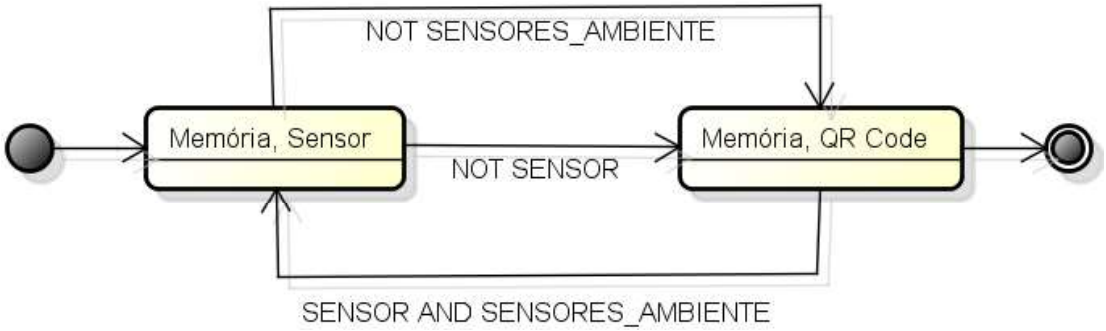

Figura 3. Máquina de estados criada na prova de conceito

Com a máquina de estados, escolhe-se um estado e são então selecionados, a partir dos cenários globais do produto, os cenários de teste do produto específicos da configuração. Ao escolher o estado "Memória e Sensor", por exemplo, são selecionados seis cenários de testes que envolvem a *feature* "Memória" ou a *feature* "Sensor". Logo, seriam os cenários de testes ***CT 01***, ***CT 03***, ***CT 05***, ***CT 06***, ***CT 07*** e ***CT 09***.

4. Avaliação preliminar

Para avaliar o método, decidiu-se por executar uma avaliação preliminar por meio de um experimento controlado (Wohlin et al., 2000). Seis voluntários participaram dessa avaliação, sendo dois alunos de graduação e quatro alunos de pós-graduação em Ciência da Computação da Universidade Federal do Ceará (UFC).

O propósito da avaliação preliminar conduzida foi avaliar, com respeito ao esforço e aplicabilidade, do ponto de vista do pesquisador, no contexto de estudantes de Ciência da Computação, o impacto do uso do ChAPTER na geração de testes. O objetivo deste experimento era então responder a seguinte questão de pesquisa: *o uso do ChAPTER promove uma maior cobertura de testes?*

A primeira atividade do experimento foi a aplicação de um questionário para caracterizar o perfil dos participantes. Os resultados deste questionário revelaram que, dentre outras coisas, os quatro alunos de pós-graduação conheciam as definições básicas de LPS, LPSSC, descrições textuais de casos de uso e teste de software, mas nenhum deles tinha estudado algo relacionado ao teste de aplicações sensíveis ao contexto. Quanto aos dois alunos de graduação, ambos conheciam o básico de teste de software e afirmaram não ter estudado nada sobre descrições textuais de casos de uso ou teste de aplicações sensíveis ao contexto.

Em seguida, com o objetivo de minimizar os efeitos da falta de conhecimento sobre os fatores do estudo experimental, foi conduzido um treinamento explorando os conceitos importantes sobre LPS, LPSSC e descrições textuais de caso de uso, bem como a estrutura do CAPLUC e como aplicar o método ChAPTER usando a ferramenta de suporte, a ToChAPTER.

Neste experimento, os voluntários foram divididos em dois grupos (A e B), ambos com dois alunos de pós-graduação e um de graduação. Cada membro do Grupo A planejou um conjunto de casos de testes com base na experiência individual para dois

casos de uso descritos textualmente da linha Mobiline. Já os participantes do Grupo B realizaram a mesma atividade utilizando a ferramenta e o método ChAPTER.

Cabe ressaltar que o aluno de graduação do Grupo B desistiu do experimento antes de completar a atividade solicitada. Já os outros dois voluntários do Grupo B haviam descrito para cada um dos 12 diferentes cenários de testes gerados pela ToChAPTER, um respectivo caso de teste. Além disso, esses voluntários relacionaram para cada um dos oito estados do produto um conjunto de casos de testes.

Na Tabela 4 são apresentados os testes planejados pelos voluntários que não utilizaram o método (Grupo A). Os Voluntários 1 e 2 são alunos de pós-graduação e o Voluntário 5 é o aluno de graduação. Nessa tabela estão descritos quais dos cenários de testes gerados pela ferramenta eles planejaram casos de testes, bem como a cobertura atingida por esses casos de testes considerando os 12 diferentes casos de testes sugeridos pela ferramenta como sendo uma cobertura de 100%.

Tabela 4. Resultados dos participantes sem o método ChAPTER

Grupo A	Tempo (min)	Testes	Cobertura (%)
Voluntário 1	38	1,2,3,7,8,9,12	58,33
Voluntário 2	31	1,2,5,8,11	41,67
Voluntário 5	24	1,2,5,8,9,10	50

Conforme observado na Tabela 4, o voluntário com melhor participação conseguiu uma cobertura de apenas 58% se comparado aos testes gerados pela ferramenta. Além disso, vale ressaltar que o voluntário do Grupo B com melhor tempo gastou 17 minutos para especificar os 12 casos de testes enquanto que o voluntário com melhor resultado do grupo A (voluntário 1) planejou 7 casos de testes em 38 minutos.

Dessa forma, os resultados da avaliação preliminar revelam indícios de que o método possibilita uma cobertura maior dos casos de testes possíveis, enquanto que o uso do método e da ferramenta de apoio pode auxiliar a reduzir o tempo gasto para o planejamento dos testes. Contudo, vale ressaltar que é necessário um experimento com um número maior de voluntários e uma diversidade maior de casos de uso para confirmar os resultados obtidos.

5. Considerações finais

Uma Linha de Produto de Software maximiza os benefícios do reuso durante o desenvolvimento de várias aplicações de um mesmo domínio. No caso de uma LPS voltada para o desenvolvimento de aplicações sensíveis ao contexto, o desafio é lidar ao mesmo tempo com as peculiaridades da linha e dos produtos finais, que usam informações de contexto.

Neste cenário, este artigo apresentou um método de geração de cenários de testes para linhas de produto de software sensíveis ao contexto, a partir de especificações textuais de casos de uso. O método, denominado de ChAPTER, gera cenários de testes em três etapas e atua na Engenharia de Domínio e na Engenharia de Aplicação. Para dar suporte à aplicação do método, uma ferramenta também foi desenvolvida. Por fim, para avaliar o método proposto, um experimento controlado foi conduzido e os resultados relevaram indícios iniciais de que o método direciona uma maior cobertura dos cenários de caso de uso.

Como trabalhos futuros, pretende-se criar métricas de cobertura para guiar a geração dos testes. Além disso, pretende-se executar um experimento controlado com um número maior de voluntários, incluindo professionais da área de teste, e uma diversidade maior de casos de uso.

6. Agradecimentos

Este trabalho é resultado parcial do projeto UbiStructure suportado pelo CNPq (MCT/CNPq 14/2011 - Universal)

Referências

Ali, M. M.; Moawad, R. (2010) An Approach for Requirements Based Software Product Line Testing. In: *7th International Conference on Informatics and Systems*, p. 1 –10.

Almeida, E. S.; Alvaro, A.; Garcia, V. C.; Mascena, J. C. C. P.; Burégio, V. A. A.; Nascimento, L. M; Lucrédio, D.; Meira, S. L. (2007) *C.R.U.I.S.E: Component Reuse in Software Engineering*. C.E.S.A.R e-book, Brasil.

Bertolino, A.; Gnesi, S. (2003) Use Case-based Testing of Product Lines. In: *9th European Software Engineering Conference*. New York, NY, USA: ACM, p. 355–358.

Cai, X.; Zeng, H. (2013). Model-based Test Generation for Software Product Line. *In 12th International Conference on Computer and Information Science*, p. 347-351.

Fernandes, P.; Werner, C.; Teixeira, E. (2011). An Approach for Feature Modeling of Context-Aware *Software* Product Line. *In Journal of Universal Computer Science*, v. 17, n. 5, mar, p. 807-829.

Northrop, L.M. (2002) SEI's *Software* Product Lines Tenets. *IEEE Software*, v.19, n.4, p.32-40.

Lamancha, B. P.; Usaola, M. P.; Velthuis, M. P. (2009) Software Product Line Testing – A Systematic Review. *In International Conference in Software and Data Technologies*, p. 23-30.

Lima Neto, C. R.; P. A. M. S.; Almeida, E. S.; Meira, S. R. L. (2012) A Software Product Lines System Test Case Tool and its Initial Evaluation. *In 13th International Conference on Information Reuse and Integration*, p. 25-32.

Ostrand, T. J.; Balcer, M. J. (1988) The Category-partition Method for Specifying and Generating Fuctional Tests. *Communications of the ACM*, USA, v. 31, n. 6, p. 676–686

Santos, I. S.; Santos Neto, P.; Andrade, R. M. C. (2013). A Use Case Textual Description for Context Aware SPL Based on a Controlled Experiment. *In Conference on Advanced Information Systems Engineering (CAiSE) Fórum*, p. 1-8.

Wang, Z.; Elbaum, S.; Rosenblum, D. S. (2007). Automated Generation of Context-Aware Tests. *In 29th International Conference on Software Engineering* (ICSE '07). IEEE Computer Society, Washington, p. 406-415.

Wohlin, C.; Runeson, P.; Host, M.; Ohlsson, M.; Regnell, B.; Wesslen, A. (2000) *Experimentation in Software Engineering: An Introduction,* Kluwer Academic Publishers, USA.