Técnica de Inspeção Baseada em *Checklist* para Identificação de Defeitos em Diagramas de Atividades

Rafael M. de Mello, Jobson L. Massollar, Guilherme H. Travassos

Programa de Engenharia de Sistemas e Computação (PESC), COPPE/UFRJ
Universidade Federal do Rio de Janeiro
Caixa Postal 68.511 – CEP 21.945-970 – Rio de Janeiro – RJ – Brasil

{rmaiani, jobson, ght}@cos.ufrj.br

Abstract. *The specification of contemporary applications uses to involve a large amount of information to be captured and represented. The using of use cases and activity diagrams represents a promising alternative whether these artifact´s quality can be assured. However, quality assurance is limited by the lack of software technologies that can support defects identification mainly considering the use of activity diagrams. Therefore, in this paper we introduce ActCheck, a configurable inspection technique based on checklist to support the identification of defects in software specifications described by activity diagrams. A feasibility study has indicated that the using of ActCheck allows the identification of different defects when compared with those ones from ad-hoc inspections even considering the need of reorganization of its checklists' items aiming at to reduce the inspection time.*

Resumo. *A especificação de aplicações de software contemporâneas costuma envolver uma quantidade de informação que necessita ser devidamente capturada e representada. Utilizar casos de uso e diagramas de atividades representa uma alternativa viável desde que se possa garantir sua qualidade. Entretanto, esta garantia está limitada pela carência atual de técnicas que apoiem a identificação de defeitos, principalmente, em diagramas de atividades. Desta forma, neste artigo apresentamos ActCheck, uma técnica configurável de inspeção baseada em checklist desenvolvida para apoiar a identificação de defeitos em diagramas de atividades usados na especificação de requisitos. Um estudo de viabilidade indicou que inspeções com ActCheck permitem identificar defeitos diferentes dos detectados por inspeções ad-hoc, apesar da necessidade de reorganização dos itens dos checklists visando otimizar o tempo de inspeção.*

1. Introdução

Na engenharia de software, o detalhamento das funcionalidades e dos processos de negócio de um projeto costuma ser realizado através de uma simples redação ou mesmo através de uma descrição textual apoiada por padrões, como a descrição através de casos de uso. Entretanto, a especificação de requisitos de aplicações contemporâneas, tais como aplicações web, usualmente envolve uma grande quantidade de informação. Tal informação pode estar relacionada aos fluxos de trabalho ou *workflows*, bem como às dependências entre esses fluxos, desvios ou mesmo regras de negócio, contexto este que pode dificultar o entendimento do escopo do projeto, quando a informação está organizada através de uma descrição meramente textual.

Gross e Doer (2009) apontam que a modelagem de processos, apoiada por notações como o diagrama de atividades (OMG, 2009), possui um papel relevante na

engenharia de requisitos. Gutiérrez et al. (2008) afirmam que a aplicação de diagramas de atividades da UML na descrição de casos de uso permite especificar o comportamento global destes casos de uso. Neste sentido, Pastor et al. (2001), Almendros-Jimenes e Iribarne (2005), Pereira et al. (2009) e Massollar (2011) propõem diferentes técnicas que fazem uso do diagrama de atividades na especificação de requisitos através dos constructos existentes no modelo de atividades da UML e estereótipos específicos para descrever, por exemplo, casos de uso.

Desde a publicação da primeira versão da UML em 1998, esta tem se destacado como padrão para modelagem de projetos orientados a objeto, provendo modelos para a representação das características estática e dinâmica de sistemas. Com o advento da versão 2.0 da UML em 2003 (OMG, 2009), o diagrama de atividades foi aprimorado e tornou-se um modelo para representação dinâmica mais abrangente, deixando de ter sua aplicação limitada a projetos de sistemas orientados a objeto e passando a ser, inclusive, adotado como notação para representação de *workflows* em geral (Wynn et al., 2009).

Na última década, também se percebe o esforço significativo da comunidade científica no aprimoramento de atividades que promovam a garantia da qualidade do produto final de software, com foco na detecção de defeitos nos artefatos elaborados ao longo do processo de desenvolvimento, como no caso de técnicas de inspeção (Wong, 2006). Devido ao potencial de redução de esforço em projetos de software, diferentes técnicas de inspeção, mais especificamente para apoiar a detecção de defeitos em especificações de requisitos (Shull et al., 2000; Belgamo e Fabbri, 2004; Mafra e Travassos, 2006) e modelos iniciais de projetos de software (Travassos et al., 1999; Barcelos e Travassos, 2006) têm sido desenvolvidas e aplicadas. Entretanto, uma *quasi*-revisão sistemática da literatura conduzida recentemente (Mello et al., 2010) não identificou tecnologias que explorassem suficientemente a inspeção de diagramas de atividades e de outros modelos para especificação de *workflow*, apesar da literatura técnica apresentar algumas tecnologias que apoiem a verificação sintática destes modelos (Eshuis e Wieringa, 2004; Choi e Watanabe, 2005; Guelfi e Mammar, 2005; Takaki et al., 2007).

Portanto, a inexistência de técnicas de inspeção específicas a diagramas de atividades da UML 2, combinada com a oportunidade de utilizar estes diagramas para a especificação de requisitos de aplicações contemporâneas, motivaram esta pesquisa. Os componentes do diagrama de atividades e suas possibilidades de aplicação foram analisados, sendo identificadas 92 situações possíveis de defeito, chamadas de casos discrepantes (Mello et al., 2010). Estes casos discrepantes, por sua vez, serviram de base para a elaboração de 34 itens de avaliação organizados em um *checklist* configurável para a inspeção de diagramas de atividades na especificação de requisitos. Esta técnica de inspeção foi então submetida a uma prova de conceito, quando foram identificadas algumas vantagens na capacidade de identificação de defeitos, quando comparadas com inspeções *ad-hoc* (Mello et al. 2010). Com a aplicação da prova de conceito, foram observadas oportunidades de melhoria na técnica, cuja estrutura sofreu, então, significativas modificações. Esta nova versão da técnica de inspeção recebeu o nome de ActCheck, que será apresentada neste artigo juntamente com os resultados de seu primeiro estudo de viabilidade.

2. ActCheck

ActCheck representa uma técnica para inspeção individual de diagramas de atividades, mais especificamente de diagrama de atividades que compõem a especificação de requisitos de projetos de software, abrangendo todos os recursos sintáticos previstos na UML 2 para este diagrama (OMG, 2009). É uma técnica configurável baseada em *checklist*, permitindo que itens de avaliação de seus dois *checklists* (A e B) sejam aplicados ou não, conforme o contexto de cada projeto, como pode ser visualizado no diagrama da Figura 1, que representa o processo de aplicação de ActCheck.

A configuração dos *checklists* deve ser realizada durante a fase de planejamento da inspeção. Com a equipe de inspetores definida, o desenvolvedor responsável pela aplicação deverá responder um questionário, para caracterização da aplicação conforme a técnica de modelagem adotada no projeto. Com base nas respostas do Questionário, o moderador da inspeção deverá configurar os *checklists* A' e B', ou seja, identificar os itens de avaliação dos *checklists* A e B de ActCheck que serão utilizados na inspeção em questão, com o auxílio de Tabelas de rastreabilidade destes dois *checklists*. Concluída a fase de planejamento, cada inspetor realizará sua inspeção individual, preenchendo os *checklists* A' e B' e informando as discrepâncias identificadas individualmente em um relatório de discrepâncias. As subseções a seguir apresentam brevemente os artefatos que compõem ActCheck.

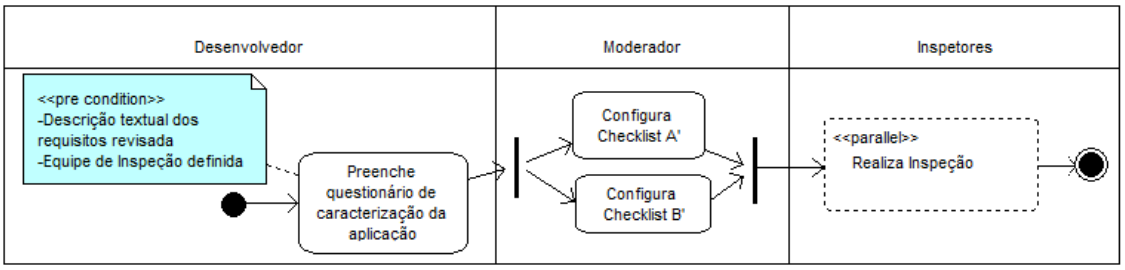

Figura 1. O processo de aplicação de ActCheck

2.1. Checklist A

O Checklist A de ActCheck é composto por 39 itens de avaliação que orientam o inspetor a identificar discrepâncias entre os diagramas de atividades de uma especificação de requisitos e a descrição textual destes requisitos. Os itens de avaliação do Checklist A independem de como a descrição textual dos requisitos foi feita, embora valha ressaltar que a aplicação de técnicas estruturadas como as propostas por Pereira et al. (2009) e Massollar (2011) e possam facilitar a identificação do escopo de cada inspeção a partir da rastreabilidade entre os artefatos da especificação. Buscando facilitar a inspeção, os itens de avaliação do Checklist A foram agrupados conforme seus respectivos focos de verificação, conforme pode ser visualizado na Tabela 1.

Tabela 1. Grupos de verificação do Checklist A de ActCheck

Grupo	# de Itens	Finalidade
Ações e condições	8	verificar individualmente cada ação e as possíveis condições existentes na atividade, sejam estas ligadas as próprias ações (pré e pós-condições locais) ou a outros construtos (como condições de guarda e critérios de repetição de regiões de expansão).
Fluxo de Controle	7	verificar se os possíveis caminhos que podem ser percorridos do início ao fim de uma atividade estão corretos, completos e consistentes com os requisitos.
Objetos e fluxo de dados	7	detectar discrepâncias no fluxo de dados do diagrama de atividades, representado através dos nós de objeto e de suas respectivas propriedades.
Raias	3	verificar se a utilização do recurso sintático de raias (*swimlanes*) está adequada ao contexto da atividade.
Estereótipos para casos de uso	14	detectar discrepâncias em diagramas de atividades de projetos que adotem a técnica de modelagem apresentada por Massollar (2011) para descrição de casos de uso

A Tabela 2 apresenta exemplos de itens de avaliação do Checklist A para cada grupo: ações e condições (1,3 e 7), fluxo de controle (9, 10 e 11), objetos (18, 21 e 22), raias (23 e 24) e estereótipos para casos de uso (28, 34, 35 e 39). Importante observar que, para cada grupo do Checklist A, os itens de avaliação foram organizados de modo a orientar o inspetor a buscar, nesta ordem, por discrepâncias referentes às seguintes categorias: omissões, inadequações (ambiguidades, inconsistências e fatos incorretos) e informações estranhas. Esta categorização é baseada na categorização introduzida por Shull et al. (1999) e Travassos et al. (1999b) para defeitos em modelos de software, que se basearam na categorização de Basili et al. (1996) para defeitos em requisitos.

Tabela 2. Exemplos de itens de avaliação do Checklist A de ActCheck

#	Item de Avaliação	Resposta
1	Todos os rótulos das ações da atividade informam qual é a ação e qual é o objeto de cada ação?	( ) Sim ( ) Não ( ) N.A.
3	As ações e condições da atividade estão em conformidade com o especificado nos requisitos?	( ) Sim ( ) Não ( ) N.A.
7	Alguma ação ou condição da atividade contém informação desnecessária?	( ) Sim ( ) Não ( ) N.A.
9	Alguma ação ou condição deixou de ser inserida na atividade, numa região ou num nó protegido?	( ) Sim ( ) Não ( ) N.A.
10	Algum nó de fim de atividade ou de fim de fluxo deixou de ser informado?	( ) Sim ( ) Não ( ) N.A.
11	As regiões de expansão, de interrupção, nós de loop, nós condicionais e as regiões protegidas (exceções) descrevem suas ações de maneira coerente com a especificação?	( ) Sim ( ) Não ( ) N.A.
18	Os objetos de entrada e de saída de cada ação, região de expansão ou subatividade estão em conformidade com o especificado nos requisitos?	( ) Sim ( ) Não ( ) N.A.
21	Todas as transformações entre objetos estão claramente descritas?	( ) Sim ( ) Não ( ) N.A.
22	Algum objeto da atividade, propriedade de objeto ou transformação de objeto apresenta informação desnecessária/irrelevante para o contexto da atividade?	( ) Sim ( ) Não ( ) N.A.
23	Os fluxos agrupados em cada raia/ sub-raia da atividade estão em conformidade com o especificado nos requisitos?	( ) Sim ( ) Não ( ) N.A.
24	As raias e sub-raias da atividade estão claramente descritas?	( ) Sim ( ) Não ( ) N.A.
28	As ações referentes ao comportamento interno do sistema estão devidamente representadas através do estereótipo <<system action>>?	( ) Sim ( ) Não ( ) N.A.
34	Cada condição de guarda ou a ação que precede uma resposta do sistema <<system response>> está relacionada a uma ação do ator?	( ) Sim ( ) Não ( ) N.A.
35	Toda ação da atividade está descrita num nível de abstração adequado para a descrição de um caso de uso?	( ) Sim ( ) Não ( ) N.A.
39	Alguma ação da atividade detalha parte da solução computacional (como é feito)?	( ) Sim ( ) Não ( ) N.A.

2.2. Checklist B

O Checklist B é composto por 14 itens de avaliação, sendo que os 11 primeiros itens orientam o inspetor a verificar a consistência interna do diagrama de atividades, categorizada como verificação intradiagrama, enquanto que os outros três itens orientam a detecção de defeitos entre diagramas, categorizada como verificação interdiagramas, conforme citado em (Tanriöver e Bilgen, 2007). A verificação intradiagrama representa uma nova oportunidade de identificação de discrepâncias na descrição textual dos

requisitos, já que defeitos eventualmente ignorados na inspeção prévia da descrição textual dos requisitos tendem a não ser também capturados através da aplicação do Checklist A, que considera a descrição textual dos requisitos como correta (oráculo). A elaboração de itens de verificação intradiagrama e interdiagramas baseia-se no entendimento de que, assim como a elaboração de diagramas de atividades pode contribuir para uma melhor compreensão dos requisitos descritos textualmente, a detecção de defeitos nesta descrição textual também poderia ser facilitada através do exame visual dos diagramas. A Tabela 3 apresenta uma amostra dos itens de avaliação do Checklist B de ActCheck.

Tabela 3. Exemplos de itens de avaliação do Checklist B de ActCheck

#	Item de Avaliação	Resposta
1	As ações e condições da atividade estão consistentes entre si?	( ) Sim ( ) Não ( ) N.A.
3	Existem ações sendo executadas concorrentemente, mas que são executadas sequencialmente em outras partes do modelo?	( ) Sim ( ) Não ( ) N.A.
4	Existem ações semelhantes/ idênticas sendo executadas em raias ou sub-raias distintas?	( ) Sim ( ) Não ( ) N.A.
7	As propriedades dos objetos estão sendo respeitadas ao longo da atividade?	( ) Sim ( ) Não ( ) N.A.
10	De acordo com o seu conhecimento do domínio, todas as condições da Atividade são viáveis?	( ) Sim ( ) Não ( ) N.A.
12	Alguma ação corresponde à outra atividade do projeto, mas esta referência não está explícita no Diagrama de Atividades?	( ) Sim ( ) Não ( ) N.A.
14	Todas as outras atividades referenciadas nas ações do Diagrama de Atividades existem no projeto?	( ) Sim ( ) Não ( ) N.A.

2.3. Questionário de Caracterização e Tabelas de Rastreabilidade

Dos 53 itens de avaliação dos *checklists* de ActCheck, apenas 10 podem ser considerados aplicáveis a qualquer diagrama de atividades, considerando-se que a forma mais simples de descrever uma atividade consiste numa sequência sem desvios de uma ou mais ações entre um nó de início e um nó de fim de atividade. Para apoiar a identificação de quais itens de ActCheck serão aplicados na inspeção dos diagramas de determinado projeto de software, foi elaborado um questionário de caracterização da aplicação. Espera-se que este questionário seja preenchido pelo desenvolvedor responsável pelo projeto, devido à necessidade de conhecimento prévio da técnica de modelagem que foi aplicada aos diagramas de atividades que serão inspecionados. O questionário de caracterização da aplicação pode ser visualizado na Figura 2.

1) Quais dos seguintes recursos do Diagrama de Atividades estão previstos na técnica de modelagem adotada na especificação de requisitos?
( ) I- Nó de decisão/ merge
( ) II- Nó de bifurcação/ sincronização
( ) III- Raias (*swimlanes*)
( ) IV- Nós de objeto
( ) V- Grupos de Atividades
( ) VI- Interrupções e Exceções
( ) VII- Regiões de Expansão, Nós de Loop e Nós Condicionais
( ) VIII- Ações de Envio e Recebimento de Sinal
( ) IX- Chamadas de atividade
( ) X- Estereótipos para descrição de casos de uso
( ) Outros: _____

2) Existe mais de um diagrama de atividades na especificação de requisitos? ( ) Sim ( ) Não

3) A técnica de modelagem adotada no projeto permite que os diagramas de atividades da especificação de requisitos representem uma mesma atividade em diferentes níveis de detalhamento ? ( ) Sim ( ) Não

4) Algum inspetor possui conhecimento do domínio da aplicação? ( ) Sim ( ) Não

Figura 2. Questionário de Caracterização da Aplicação

É importante destacar que as respostas ao questionário devem considerar o que está *previsto* na técnica de modelagem, e não somente aquilo que é *aplicado* nos diagramas de atividades de um projeto. Por exemplo, pode estar prevista a utilização de chamadas de subatividades e o desenvolvedor não identificar nenhuma chamada de atividade em um diagrama, embora fosse necessário fazê-lo, caracterizando uma omissão.

Com o questionário de caracterização da aplicação devidamente preenchido, são consultadas as Tabelas de rastreabilidade de cada *checklist* de ActCheck para configurar os *checklists* da inspeção. Tais Tabelas indicam quais são as respostas esperadas no questionário para que os itens de avaliação façam parte dos *checklists* de uma inspeção, conforme pode ser visualizado na Tabela 4, que apresenta parte da Tabela de rastreabilidade do Checklist B.

Tabela 4. Tabela de Rastreabilidade do Checklist B

#	Resposta Esperada	Classes de Defeito Relacionadas	Item do Questionário de Caracterização da Aplicação
5	Sim	Inconsistência	1) VII
6	Sim	Inconsistência	1) IV
7	Sim	Inconsistência	1) IV
8	Sim	Ambiguidade	1) VIII
9	Sim	Fato Incorreto, Inconsistência	1) II e 4) Sim
10	Sim	Fato Incorreto, Inconsistência	4) Sim
11	Sim	Fato Incorreto, Inconsistência	1) VIII e 4) Sim

Tendo sido apresentada a técnica de inspeção, a seguir apresentaremos um estudo experimental *in vitro* conduzido para avaliar a sua viabilidade, descrevendo os resultados obtidos.

3. Estudo Experimental

O processo de desenvolvimento e experimentação de ActCheck segue a abordagem baseada em evidências para definição de novas tecnologias de software, apresentada em (Mafra et al., 2006). Tal abordagem consiste numa extensão da proposta de Shull et al. (2001) para introdução de tecnologias de software na indústria, aplicada em pesquisas realizadas pelo grupo Engenharia de Software Experimental da COPPE/UFRJ.

Inicialmente, para que a definição de novas tecnologias esteja baseada em evidências da literatura, devem ser executados estudos secundários, mais precisamente revisões sistemáticas, conforme o estudo secundário referente a esta pesquisa apresentado em (Mello et al., 2010). Em seguida, é elaborada a versão inicial da tecnologia, baseando-se nos resultados obtidos nos estudos secundários. Com a versão inicial da tecnologia definida, são conduzidos os seguintes estudos primários, nesta ordem: estudos de viabilidade, estudos de observação, estudos de caso em um ciclo de vida de desenvolvimento real e estudos de caso na indústria.

Neste sentido, nos meses de novembro e dezembro de 2010 foi conduzido um estudo experimental *in vitro* para avaliar a viabilidade de ActCheck. O objetivo geral deste estudo consistiu na avaliação dos efeitos da aplicação de ActCheck na documentação parcial do projeto de uma aplicação real, no caso o módulo financeiro (MFI) de um sistema de informação baseado na Web, desenvolvido pelo Grupo de Engenharia de Software Experimental para uma fundação de apoio a projetos de

pesquisa e desenvolvimento no Rio de Janeiro. Além de avaliar a viabilidade da técnica, o estudo também envolveu a comparação entre os resultados obtidos nas inspeções com ActCheck e os resultados de inspeções *ad-hoc*. A Figura 3 apresenta os objetivos específicos definidos para o estudo.

Analisar: os efeitos da aplicação de ActCheck
Com o propósito de: caracterizar
Em relação à: sua viabilidade (opinião dos participantes, eficácia e eficiência na identificação de defeitos)
Do ponto de vista de: inspetores de requisitos
No contexto de: inspeções de diagramas de atividades descrevendo casos de uso do módulo financeiro de um sistema de informação Web de larga escala por estudantes de pós-graduação de Engenharia de Software.

Figura 3. Objetivos específicos do estudo experimental.

A documentação utilizada nas inspeções foi composta pela especificação de requisitos (textual) previamente revisada do MFI e por um conjunto de diagramas de atividades, descrevendo seus casos de uso. A descrição dos requisitos a partir dos diagramas de atividades foi feita com base na abordagem apresentada em Massollar (2011), que utiliza um subconjunto de recursos sintáticos do diagrama de atividades acrescido de estereótipos específicos para descrição de casos de uso.

3.1 Planejamento

O estudo buscou responder às seguintes questões de pesquisa:

1. A aplicação de ActCheck auxilia na detecção de defeitos?
2. O tempo utilizado na inspeção com ActCheck é bem aplicado?
3. Uma inspeção aplicando ActCheck é mais eficaz do que uma inspeção *ad-hoc*?
4. ActCheck estimula os inspetores à sua aplicação?

Para apoiar a resposta às três primeiras questões, foram consideradas as seguintes métricas básicas: quantidade de defeitos, quantidade de discrepâncias e tempo dedicado à inspeção. A partir destas métricas, foram derivadas a eficácia (razão entre a quantidade de defeitos detectados e o total de defeitos conhecidos) e a eficiência (razão entre a quantidade de defeitos detectados e o tempo dedicado a uma inspeção). Para responder à quarta questão, também foi considerada a opinião dos participantes do estudo sobre ActCheck. Deste modo, considera-se que este estudo é quantitativo, por contabilizar resultados para posterior verificação da eficiência e da eficácia da técnica, mas também qualitativo, por abordar a opinião dos participantes. A partir das questões de pesquisa, foram formuladas duas hipóteses nulas (H01 e H02) e suas hipóteses alternativas correspondentes (HA1 e HA2), conforme observado na Figura 4:

H01: Não há diferença entre a eficiência de inspeções de diagramas de atividades que aplicam ActCheck e a eficiência de inspeções *ad-hoc*.
HA1: A eficiência de inspeções de diagramas de atividades que aplicam ActCheck é maior que a eficiência de inspeções *ad-hoc*.
H02: Não há diferença entre a eficácia de inspeções de diagramas de atividades que aplicam ActCheck e a eficácia de inspeções *ad-hoc*.
HA2: A eficácia de inspeções de diagramas de atividades que aplicam ActCheck é maior que a eficácia de inspeções *ad-hoc*.

Figura 4. Hipóteses do estudo experimental

O estudo contou com a participação de oito alunos de pós-graduação (7 mestrandos e 1 doutorando) do curso de VV&T (Verificação, Validação e Testes) oferecido pelo Programa de pós-graduação em Engenharia de Sistemas da Computação da COPPE/UFRJ. Cada participante assinou um termo de consentimento para participar

deste estudo e preencheu um formulário de caracterização do participante. Estes oito participantes foram organizados em quatro duplas com nível de experiência gradual, conforme o grau de experiência relatado para temas relacionados à engenharia de software, em especial inspeção de software, diagrama de atividades e especificação de requisitos. Os participantes deste estudo, no entanto, possuíam ao menos experiência teórica e prática em inspeções de software, tendo atuado anteriormente em sessões de inspeção, aplicando técnicas diferenciadas (*ad-hoc*, baseada em heurísticas, *checklists* e técnicas de leitura), todas explorando taxonomia de defeitos equivalente. Cada dupla possuía um participante com conhecimento do domínio da aplicação.

Para a realização do estudo, foram selecionados quatro casos de uso de complexidade variada (6, 15, 36 e 40) do MFI, que já haviam sido inspecionados e implementados pela equipe do projeto. Estes casos de uso, por sua vez, foram descritos por duas versões de diagrama de atividades construídos por membros da equipe do projeto: uma versão modelada de modo *ad-hoc* utilizando-se a ferramenta BoUML, e outra versão modelada com apoio de um *plug-in* específico para o BoUML, desenvolvido para apoiar modelagem de casos de uso através de diagramas de atividades (Massollar, 2011) e reduzir defeitos. Cada participante recebeu a tarefa de realizar, em cada rodada, duas inspeções individuais de dois diagramas de atividades, sem repetir nenhum diagrama, conforme a distribuição contida na Tabela 5, que mostra também que, ao fim das duas rodadas, todos os diagramas foram inspecionados por todos os participantes. Na primeira rodada, os participantes deveriam realizar inspeções *ad-hoc*, enquanto que, na segunda rodada, deveriam aplicar ActCheck.

Tabela 5. Distribuição dos artefatos entre os participantes para cada rodada.

Dupla	Nível de Experiência	Participante	Primeira Rodada (*ad-hoc*)		Segunda Rodada (ActCheck)	
			sem plug-in	*com plug-in*	*sem plug-in*	*com plug-in*
A	Alto	P1	15	40	6	36
		P2	6	36	15	40
B	Médio-Alto	P3	36	40	6	15
		P4	6	15	36	40
C	Médio-Baixo	P5	36	6	40	15
		P6	40	15	36	6
D	Baixo	P7	15	6	40	36
		P8	40	36	15	6

3.2 Preparação

Antes da execução de cada rodada, foram realizados treinamentos específicos para preparação dos participantes. A preparação para a primeira rodada contou com uma exposição de 1 hora e 30 minutos, que teve como objetivo principal disseminar e nivelar o conhecimento dos participantes quanto ao diagrama de atividades, sendo apresentada também a técnica utilizada para descrição de casos de uso (Massollar, 2011) e as categorias de defeito a serem aplicadas nas inspeções (Shull et al., 1999). Antes da execução da segunda rodada, os participantes assistiram a uma exposição de 1 hora e 30 minutos sobre ActCheck, sendo apresentados exemplos de defeitos para cada item de avaliação incluído nos *checklists* do estudo, conforme previamente configurado pelos moderadores da inspeção, no caso os próprios pesquisadores.

3.3 Execução

Após a conclusão de cada treinamento, cada participante recebeu via e-mail um pacote contendo a especificação de requisitos do MFI, dois diagramas de atividades e dois relatórios de discrepâncias a serem preenchidos pelo participante, um para cada diagrama inspecionado. No caso da segunda rodada, cada participante também recebeu os dois *checklists* de ActCheck configurados. Ao fim de cada rodada, todos os participantes encaminharam seus relatos de discrepâncias conforme previsto. Deste modo, os resultados das 32 inspeções foram considerados, correspondendo às duas inspeções de cada um dos oito diagramas em cada uma das duas rodadas. Com a conclusão da segunda rodada de inspeção, os pesquisadores conduziram uma entrevista com os oito participantes, onde foram coletadas as percepções destes sobre o estudo em geral e sobre os aspectos negativos e positivos de ActCheck. Propostas para promover a melhoria da técnica também foram coletadas.

As discrepâncias relatadas pelos participantes em cada rodada foram analisadas por dois pesquisadores, extraindo-se os defeitos distintos e descartando-se os falsos positivos. Para tal, primeiramente, cada pesquisador conduziu sua análise individual das discrepâncias. Em seguida, através de uma reunião, os dois pesquisadores confrontaram suas análises até chegar a um consenso sobre cada item divergente. Um terceiro pesquisador acompanhou e revisou todo o processo.

3.4 Análise Quantitativa

Para avaliar as hipóteses do estudo, os resultados obtidos nas rodadas foram submetidos a análises apoiadas pela ferramenta estatística JMP4 (SAS, 2011), onde se buscou avaliar também outros aspectos relacionados às inspeções, como o comportamento da variável tempo e da variável quantidade de defeitos. A distribuição dos dados foi avaliada quanto à normalidade e a significância estatística dos resultados coletados foi verificada considerando-se o teste da Distribuição de Student: média da população e diferenças entre médias, com $\alpha = 10\%$, devido ao pequeno número de participantes.

Inicialmente, foi observado que as hipóteses nulas (H01 e H02) foram refutadas, uma vez que as inspeções com ActCheck apresentaram tanto eficiências quanto eficácia distintas das inspeções *ad-hoc*. Entretanto, os testes das hipóteses alternativas (HA1 e HA2) apontaram que não há significância estatística suficiente na amostra apresentada para afirmar que inspeções com ActCheck são mais ou menos eficazes/ eficientes do que as inspeções *ad-hoc*. A Figura 5-a representa a distribuição dos dados por inspeção individual realizada em cada rodada (*ad-hoc* e ActCheck) a fim de verificar a hipótese alternativa HA1, referente à eficiência, enquanto que a Figura 5-b representa a mesma distribuição a fim de verificar HA2 (eficácia), já considerando a remoção de um *outlier* para cada caso.

Entretanto, observou-se uma diferença estatisticamente significativa na variável tempo entre as inspeções *ad-hoc* e as inspeções com ActCheck, sendo que ActCheck, na média, exige mais tempo do inspetor, como pode ser observado na Figura 6, onde a dimensão do tempo apresenta-se em minutos e cada ponto do gráfico corresponde ao tempo que um participante levou para realizar suas inspeções em uma rodada. Em contrapartida, não é possível afirmar que inspeções com ActCheck propiciam a identificação de mais defeitos do que inspeções *ad-hoc*.

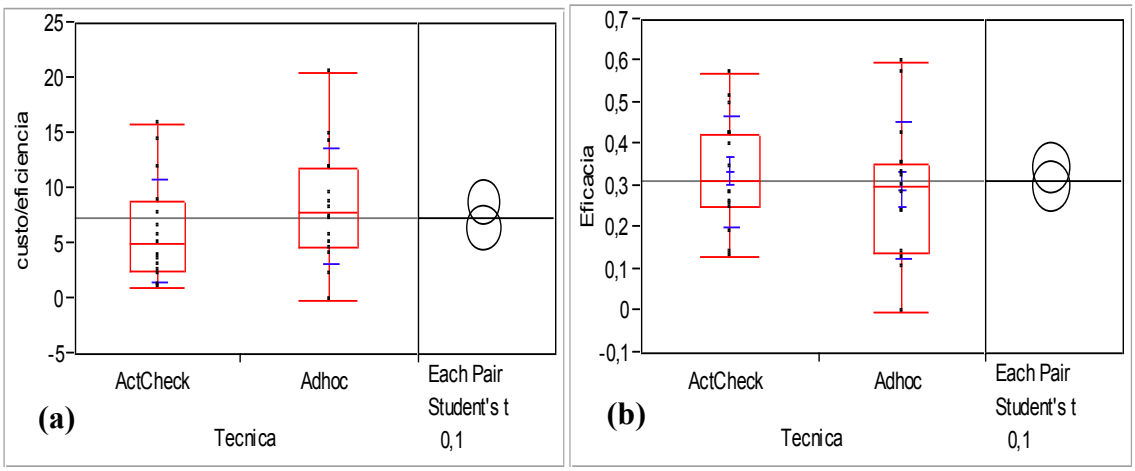

Figura 5. Distribuição da eficiência (a) e da eficácia (b) para ActCheck e *ad-hoc* com teste de Student para cada distribuição (α = 10%).

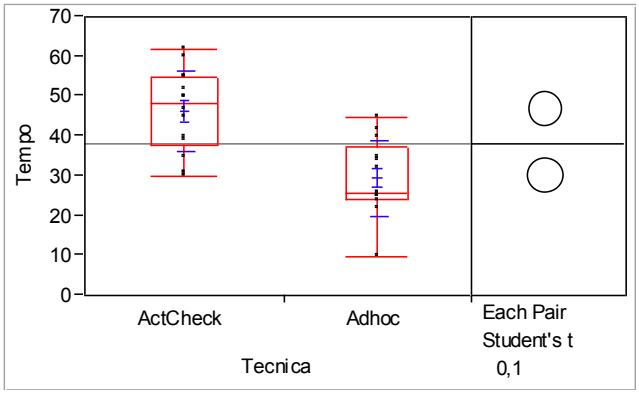

Figura 6. Distribuição do tempo para ActCheck e *ad-hoc* por inspetor com teste de Student para cada distribuição (α = 10%).

Análises estatísticas adicionais foram feitas comparando os resultados entre os grupos e entre os participantes com e sem conhecimento prévio do domínio do problema. Entretanto, não foi possível afirmar que ActCheck tenha promovido alguma mudança significativa na eficácia ou na eficiência em algum destes agrupamentos. Para buscar compreender as causas das hipóteses alternativas não terem sido confirmadas, foi conduzida uma investigação mais detalhada sobre os dados coletados, a fim de identificar as possíveis oportunidades para melhoria. A Tabela 6 apresenta um resumo dos resultados coletados nas duas rodadas de inspeção, indicando que alguns defeitos ainda não estão sendo contemplados por ActCheck.

Tabela 6. Resumo dos dados coletados nas duas rodadas de inspeção do estudo.

Item observado	Rodada 1 (*ad-hoc*)	Rodada 2 (ActCheck)
Número de Discrepâncias detectadas	132	121
Total de Falsos Positivos	48	24
Total de defeitos detectados	84	97
Tempo total dedicado às inspeções (minutos)	654	944
Eificência (defeitos/hora)	7,71	6,17
Total de defeitos distintos	75	88
Defeitos distintos na rodada/ total de defeitos	60%	70%
Eficácia média de cada inspeção	32,9%	33,4%

Analisando os dados da Tabela 6, percebe-se que a aplicação de ActCheck reduziu em 50% a quantidade de falsos positivos das inspeções (de 48 para 24) embora pouco

tenha contribuído para a detecção de mais defeitos (aumento de 15% no total de defeitos da rodada). Deste modo, ocorreu com ActCheck um ligeiro aumento na cobertura de defeitos (de 60% para 70%), embora a eficácia de cada inspeção individual tenha sido praticamente a mesma (aproximadamente 33%, sem remoção de *outliers*). Em contrapartida, com o tempo total dedicado às inspeções com ActCheck foi bem maior (aumento de 44%), houve uma relevante queda na eficiência da técnica avaliada (20%), quando esta é comparada com a eficiência das inspeções *ad-hoc* (em defeitos/hora).

Analisando-se a quantidade de defeitos detectados em comum nas duas rodadas, chamou a atenção o fato de que 46% dos defeitos totais das duas rodadas só foram detectados através de ActCheck (64 de 139), o que sugere a contribuição da técnica para a identificação de novos defeitos. Ao mesmo tempo, verificou-se também que somente menos de um terço dos defeitos detectados nas inspeções *ad-hoc* (24 de 75) também foram detectados nas inspeções com ActCheck, o que significa que 37% da quantidade total de defeitos (51 de 139) não foi detectada nas inspeções com ActCheck, conforme pode ser observado na Figura 7.

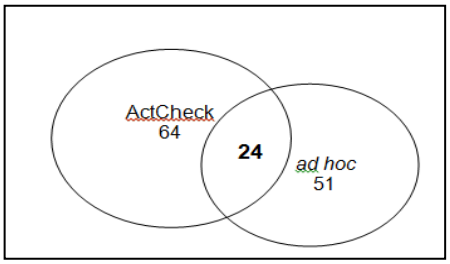

Figura 7. Quantidade de defeitos detectados somente na segunda rodada (ActCheck), em ambas as rodadas e somente na primeira rodada (*ad-hoc*).

Para entender melhor este comportamento, buscou-se identificar se cada um dos 51 defeitos detectados somente nas inspeções *ad-hoc* poderia ter sido detectado com o apoio de algum item de avaliação do Checklist A ou do Checklist B. Após esta análise, foi identificado que apenas 11 dos 51 defeitos detectados somente nas inspeções *ad-hoc* não estavam contemplados por ActCheck. Destes 11 defeitos não contemplados pelo checklist, a maior parte deles estava relacionada com a ausência de determinadas verificações referentes ao uso de regras de negócio na descrição de casos de uso.

Tabela 7- Distribuição dos defeitos detectados nas inspeções com ActCheck e defeitos detectados somente nas inspeções *ad-hoc* por item de avaliação dos *checklists* de ActCheck.

Checklist	Grupo	Defeitos detectados com ActCheck	Defeitos detectados somente com inspeções *ad-hoc*	% cobertura de ActCheck
A	Ações e Condições	22	6	79%
	Fluxo de Controle	24	26	48%
	Objetos	1	0	100%
	Casos de Uso	31	5	86%
B	-	7	3	70%
Sem item ActCheck		3	11	21%
Total		**88**	**51**	**63%**

A Tabela 7 apresenta, para cada item de avaliação, o número de defeitos detectados com ActCheck, detectados somente com inspeções *ad-hoc* e a cobertura de ActCheck (razão entre o número de defeitos detectados com ActCheck e o total de defeitos detectados). Nessa tabela é possível observar que somente o grupo "Fluxo de

Controle" do Checklist A foi responsável por mais da metade do total de defeitos não detectados na rodada de inspeções com ActCheck (26 de 51), embora os demais itens de avaliação de outros grupos também não tenham apresentado a cobertura mínima esperada, ou seja, detectar os mesmos defeitos identificados nas inspeções *ad-hoc*.

A constatação de que a maioria os defeitos somente detectados através das inspeções *ad-hoc* poderia ter sido detectada através dos itens de avaliação dos *checklists*, direcionou a análise para a qualidade destes itens, objeto de debate da análise qualitativa.

3.5 Análise Qualitativa

Em uma entrevista conduzida pelos pesquisadores, os participantes relataram suas impressões sobre o estudo, mais particularmente sobre o treinamento que receberam, sobre o material da inspeção e sobre ActCheck. Os pesquisadores incentivaram os participantes a emitirem suas opiniões, evitando direcionar os temas de discussão. Sobre os treinamentos, os participantes alegaram que, apesar de terem sido importantes para a aquisição de conhecimento, estes foram insuficientes para prepará-los para as inspeções, por abordarem um grande conteúdo de tópicos em pouco tempo. Além disto, embora ActCheck tenha sido apresentada através de diversos exemplos no segundo treinamento, os participantes observaram que estes exemplos não estavam encadeados como num processo de inspeção real e o tempo não foi suficiente para abordar adequadamente cada item de avaliação.

Como proposta de melhoria dos treinamentos, foi sugerida a inclusão de situações observadas em projetos reais. Outra sugestão observada foi a criação de um guia para direcionar o uso ActCheck, o que sugere que os participantes sentiram falta de um apoio mais sistemático ao longo das inspeções, como ocorre com uma técnica de leitura. Sobre a documentação do módulo MFI, os participantes observaram que esperavam uma documentação mais densa, pois havia pouca referência sobre o domínio do problema para detecção de defeitos nos diagramas de atividades. Como exemplo, os participantes citaram que só havia a descrição dos casos de uso no próprio diagrama de atividades. Independente do conhecimento prévio do domínio da aplicação, os participantes entenderam que possuíam pouco referencial para detectar discrepâncias e, talvez, permitir acesso a documentação mais elaborada e completa de todo o sistema poderia ter contribuído com a identificação de defeitos. Entretanto, isso não seria possível no contexto deste estudo.

Sobre ActCheck, os participantes perceberam um retrabalho muito grande na aplicação dos itens de avaliação da técnica. Para eles, a subdivisão dos itens de avaliação por categorias de defeito torna o processo cansativo e repetitivo, pois um mesmo construto teria que ser observado mais de uma vez para relatar categorias de defeito distintas em momentos distintos dos dois *checklists*. As considerações dos participantes foram muito importantes para explicar o porquê de muitos itens detectados nas inspeções *ad-hoc* não terem sido detectados nas inspeções com ActCheck. Embora os participantes tenham percebido uma cobertura mais ampla da técnica no que diz respeito à possibilidade de identificar novos defeitos, diferentes daqueles que habitualmente buscariam em inspeções *ad-hoc*, a forma como os *checklists* foram estruturados não incentivou a exploração de seus recursos. Do mesmo modo, a repetição poderia explicar o porquê das inspeções com ActCheck terem sido, na média, significativamente mais demoradas do que as inspeções *ad-hoc.*

Como aspectos positivos de ActCheck, os participantes relataram que os itens de avaliação descritos nos *checklists* são abrangentes e os levaram a detectar defeitos que muito provavelmente não observariam nas inspeções *ad-hoc*. Consequentemente, os participantes também relataram que a aplicação de ActCheck contribuiu para que estes internalizassem novas oportunidades de detecção de defeitos em diagramas de atividades. Quando interrogados se sentiam-se mais preparados para uma nova inspeção *ad-hoc* após aplicarem ActCheck, os participantes relataram que sim.

Como propostas de melhoria, os participantes da entrevista concluíram que os itens de avaliação precisam ser reorganizados, de modo a otimizar os resultados da aplicação da técnica e aproveitar melhor as oportunidades que ActCheck oferece para detecção de defeitos.

3.6 Lições aprendidas

No que diz respeito à ActCheck, foi possível extrair importantes lições que serão consideradas na evolução da técnica. No entanto, entendemos também que estas possam contribuir na elaboração de futuras técnicas de inspeção.

Primeiramente, destacamos a importância de se observar previamente como os inspetores buscam defeitos em determinado artefato de software. Esta observação poderia contribuir para que os recursos de ActCheck fossem mais bem aproveitados. Neste sentido, destacamos também que a organização dos itens de um *checklist* em subgrupos sintáticos não necessariamente contribui para melhores resultados.

Do mesmo modo, práticas como dividir os itens de avaliação em mais de um *checklist* ou mesmo utilizar itens de avaliação excessivamente padronizados e restritos a determinada categoria de defeito pode tornar as inspeções repetitivas e, consequentemente, desestimular os inspetores.

3.7 Ameaças à Validade do Estudo

Como ameaças a validade deste estudo, destacamos a pequena quantidade de participantes (oito) e também a pequena quantidade de artefatos inspecionados. A avaliação estatística por grupos, por exemplo, foi significativamente limitada, pois cada grupo possuía apenas dois participantes. Além disto, a pequena quantidade de participantes também limitou a combinação de grupos e de rodadas de inspeção, podendo causar algum viés. Entretanto, é importante ressaltar que nenhum participante inspecionou o mesmo diagrama mais de uma vez. Outra ameaça à validade consiste na especificidade do material inspecionado, limitado a um único módulo (financeiro) de um sistema real.

A inexistência de uma lista prévia de defeitos conhecidos também pode ser considerada uma ameaça à validade, afetando diretamente o cálculo da cobertura das inspeções. Foram considerados como defeitos conhecidos apenas os defeitos distintos que foram detectados em ambas as rodadas. É importante ressaltar também que os pesquisadores que avaliaram as discrepâncias encaminhadas pelos participantes pertencem ao grupo de Engenharia de Software Experimental da COPPE/UFRJ, onde um dos pesquisadores deste estudo desenvolveu ActCheck e onde também estão inseridas as atividades de projeto do MFI.

4. Considerações Finais

O estudo experimental aplicado mostrou que ActCheck é capaz apoiar a detecção de defeitos. Entretanto, a técnica precisa ser reestruturada para que seus recursos sejam mais bem explorados e o tempo de inspeção seja otimizado. Neste sentido, está sendo desenvolvida pelo grupo de Engenharia de Software Experimental uma nova versão de ActCheck, que busca tratar as principais dificuldades relatadas pelos participantes e aumentar a cobertura dos itens menos eficazes.

Deste modo, como proposta de pesquisa futura, incluímos a elaboração desta nova versão da técnica e a aplicação de um novo estudo de viabilidade, semelhante ao apresentado neste artigo e a condução de estudos observacionais e estudos de caso, conforme os novos resultados. Como alternativa, também pode ser considerada a evolução de ActCheck para uma técnica de leitura, o que ofereceria aos inspetores um recurso mais sistemático para apoiar a detecção de defeitos do que um checklist.

5. Agradecimentos

Agradecemos aos alunos do curso de VV&T (Verificação, Validação e Testes), oferecido pelo Programa de Engenharia de Sistemas da Computação da COPPE/UFRJ, pela participação no estudo. Agradecemos também ao CNPq, Fundação COPPETEC – Projeto SIGIC e ao Banco do Brasil pelo apoio a esta pesquisa.

Referências

Almendros-Jimenes, J. e Iribarne, L. (2005) "Describing Use Cases with Activity Charts", LNCS, v. 3511, Springer-Verlag Heidelberg, pp. 153-167.

Barcelos, R. F. e Travassos, G. H. (2006) "ArqCheck: Uma Abordagem para inspeção de documentos arquiteturais baseada em *checklist*", Anais do V SBQS.

Basili, V. R., Green, S., Laitenberger, O., Lanubile, F., Shull, F., Sørumgård, S. e Zelkowitz, M. (1996) "The Empirical Investigation of Perspective-Based Reading" Empirical Software Engineering: An International Journal, v. 1, n. 2, pp.133-164.

Belgamo A. e Fabbri S. (2004) "GUCCRA: Contribuindo para a identificação de defeitos em documentos de requisitos durante a construção de modelos de casos de uso", Anais do VII Workshop de Engenharia de Requisitos (WER'04), pp.100-111.

Choi E. e Watanabe H. (2005) "Model checking class specifications for web applications", Proceedings of 12th Asia-Pacific Software Engineering Conference.

Eshuis, R. e Wieringa, R. (2004) "Tool support for verifying UML activity diagrams," IEEE Transactions On Software Engineering, v. 30, n. 7.

Gross, A. e Doerr, J. (2009) "EPC vs. UML Activity Diagram - Two Experiments Examining their Usefulness for Requirements Engineering", Proceedings of 17th IEEE International Requirement Engineering Conference.

Guelfi N. e Mammar A., "A formal semantics of timed activity diagrams and its PROMELA translation", Proceedings of 12th Asia-Pacific Software Engineering Conference.

Gutiérrez J. J., Nebut C., Escalona M. J., Mejías M. e Ramos, I. M. (2008) "Visualization of Use Cases through Automatically Generated Activity Diagrams", LNCS, v. 5301, pp.83–96.

Mafra, S. N., Travassos, G. H. (2006) "Leitura Baseada em Perspectiva: A Visão do Projetista Orientada a Objetos", Anais do V SBQS.

Mafra, S. N., Barcelos, R. F., Travassos, G. H. (2006) "Aplicando uma Metodologia Baseada em Evidência na Definição de Novas Tecnologias de Software", Anais do XX SBES.

Massollar, J. (2011) "Uma Abordagem para Especificação de Requisitos Dirigida por Modelos Integrada ao Controle da Qualidade de Aplicações Web", Tese de Doutorado, Programa de Engenharia de Sistemas e Computação, COPPE/UFRJ.

Mello, R. M., Pereira, W. M., Travassos, G. H. (2010) "Activity Diagram Inspection on Requirements Specification", Anais do XXIV SBES.

OMG (2009) "OMG Unified Modeling Language (OMG UML), Superstructure-Version 2.2", http://www.omg.org/spec/UML/2.2/Superstructure.

Pastor, O., Abrahão, S., Fons, J. (2001) "An Object-Oriented Approach to Automate Web Applications Development", LNCS, v. 2115, Springer-Verlag, pp. 16-28.

Pereira, W. M., Araújo, M. A. P. e Travassos, G. H. (2009) "Apoio na Concepção de *Workflow* Científico Abstrato para Estudos *in virtuo* e *in silico* em Engenharia de Software", Proceedings of VI ESELAW.

SAS (2011) "JMP Software", http://www.jmp.com/.

Shull, F., Travassos, G. H., Carver, J., Basili, V. R. (1999) "Evolving a Set of Techniques for OO Inspections", Technical Report CSTR-4070, University of Maryland.

Shull, F., Rus, I. e Basili, V. (2000) "How perspective-based reading can improve requirements inspections", IEEE Computer, v. 33, n. 7, pp. 73-79.

Shull, F., Carver, J., Travassos, G. (2001) "An Empirical Methodology for Introducing Software Processes", Proceedings of the 8th European Software Engineering Conference - 9th ACM SIGSOFT International Symposium on Foundations of Software Engineering. ACM Press, pp. 288 – 296.

Takaki, O., Seino, T., Takeuti, I., Izumi N. e Takahashi K. (2007) "Verification Algorithm of evidence life cycles in extended UML activity diagrams", Proceedings of the International Conference On Software Engineering Advances (ICSEA'07).

Tanriöver, O. e Bilgen, S. (2007) "An inspection approach for conceptual models in notations derived from UML: a case study", 22nd International Symposium on Computer. and Information. Sciences, pp. 57-62.

Travassos, G. H., Shull, F., Fredericks, M. e Basili, V. R. (1999) "Detecting defects in object-oriented designs: using reading techniques to increase software quality". OOPSLA, v. 34, n. 10, pp. 47-56.

Travassos, G. H., Shull, F., Carver, J. (1999b) "Evolving a Process for Inspecting OO Designs", XIII SBES: Workshop de Qualidade de Software.

Wong, Y. K. (2006) "Modern Software Review- Techniques and Technologies", IRM Press, 2006.

Wynn, M. T. , Verbeek, H. M. W. , Van Der Aalst, W. M. P., ter Hofstede A. H. M. e Edmond, D. (2009) "Business process verification- finally a reality!", Business Process Management Journal, v. 15, n. 1, pp. 74-92.