Identificação Interativa da Causa Raiz de Problemas em Execuções de Mudanças de TI

Ricardo Luis dos Santos[1], **Juliano Araújo Wickboldt**[1], **Roben Castagna Lunardi**[1], **Bruno Lopes Dalmazo**[1], **Lisandro Zambenedetti Granville**[1], **Luciano Paschoal Gaspary**[1]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Porto Alegre – RS – Brasil
{rlsantos, jwickboldt, rclunardi, bldalmazo, granville, paschoal}@inf.ufrgs.br

Abstract. *The reuse of knowledge acquired by operators to diagnose failures in Information Technology (IT) infrastructures has potential to improve the process of root cause identification of recurring failures, minimizing potential losses and maintenance costs. Nevertheless, in existing solutions the diagnostic process is performed in an ad hoc and static fashion, which hampers the reuse of knowledge in recurring and similar failures. In previous work, we have proposed a solution to identify the root cause of reccuring problems in IT change management, adaptable to the current state of target infrastructure. In this paper we extend our previous this solution, improving the root cause identification process. Experiments carried out considering recurring failures provide evidence about the improvements in new version of solution compared to the previous results.*

Resumo. *O reuso do conhecimento adquirido no diagnóstico de falhas em infraestruturas de Tecnologia da Informação (TI) tem potencial para agilizar o processo de identificação da causa raiz de falhas recorrentes, minimizando possíveis perdas e custos de manutenção. Apesar disso, nas soluções existentes o processo de diagnóstico é realizado de uma forma estática e ad hoc, o que dificulta o reuso do conhecimento em falhas recorrentes ou similares. Em um trabalho anterior, propomos uma solução para identificação da causa raiz de falhas recorrentes em mudanças de TI, adaptável ao atual estado da infraestrutura alvo. Neste artigo é apresentada uma extensão da solução passada incluindo melhorias no processo de identificação da causa raiz. Experimentos conduzidos considerando falhas recorrentes evidenciam as melhorias da solução aprimorada em comparação aos resultados anteriores.*

1. Introdução

A Gerência de Serviços de Tecnologia da Informação (ITSM - *Information Technology Service Management*) tem recebido massiva atenção nos últimos anos, principalmente devido à crescente importância dos recursos e serviços de Tecnologia da Informação (TI) para a continuidade dos negócios das organizações [Sauvé *et al.* 2007]. Neste contexto, coletâneas de boas práticas e processos como o *Information Technology Infrastructure Library* (ITIL) [ITIL 2010] têm ajudado organizações a manter de forma apropriada seus ativos e serviços de TI, sendo de especial importância para aquelas caracterizadas pelos seus serviços de larga escala e altamente dinâmicos.

Entre as disciplinas descritas no ITIL, o Gerenciamento de Mudanças é aquela que define como mudanças em infraestruturas de TI devem ser planejadas, agendadas, implementadas e avaliadas. A especificação inicial de uma mudança é expressa em documentos chamados Requisições de Mudanças (RFCs - *Requests for Change*), que definem o que precisa ser mudado, mas não especificam como isto deve ocorrer. Com base nas

definições de uma RFC, um operador gera um Plano de Mudança (CP - *Change Plan*). Tal plano é essencialmente um *workflow*, composto de atividades que, uma vez implementadas, farão a infraestrutura de redes e serviços (também chamados de infraestruturas de TI ao longo do artigo) migrarem para um novo estado consistente que reflita as alterações solicitadas na RFC original [Machado *et al.* 2008] [Cordeiro *et al.* 2009].

Apesar das significativas melhorias que a adoção das boas práticas do ITIL pode proporcionar para a operação e gerência de infraestruturas de redes e serviços, a ocorrência de falhas em processos de TI não pode ser negligenciada pelos operadores e tomadores de decisão. Para lidar com eventuais problemas no processo de negócios, o ITIL propõe a disciplina de Gerenciamento de Problemas, a qual é responsável por gerir o ciclo de vida dos problemas de TI [ITIL 2007]. Seus principais objetivos são: (*i*) prevenir a ocorrência de problemas em infraestruturas de TI, (*ii*) eliminar problemas recorrentes e (*iii*) minimizar o impacto de problemas para a continuidade dos negócios. Para alcançar tais objetivos, o reuso do conhecimento e da experiência do operador em relação aos processos de TI é de fundamental importância. Pois possibilitam a simplificação de procedimentos para a detecção da causa raiz de problemas de TI e, consequentemente, reduzem os custos associados.

Para auxiliar no processo de diagnóstico de problemas, as organizações utilizam ferramentas de apoio a análise e identificação da causa raiz em infraestruturas de TI. Tais ferramentas contêm passos pré-definidos que ajudam a diagnosticar e mitigar falhas. A utilização destas ferramentas de diagnóstico, quando não identifica a causa raiz da falha, possibilita o fornecimento de informações relevantes ao pessoal técnico responsável.

Apesar de apresentarem alguma eficácia, a descrição de casos nessas soluções é feita de forma *ad hoc* (por exemplo, através de *scripts* contendo perguntas e respostas). Normalmente, esses casos são estáticos, tornando praticamente impossível o seu reuso completo ou parcial. Isto limita a identificação da causa raiz de problemas similares e, consequentemente, diminui a probabilidade de sucesso do processo de diagnóstico. Além disso, devido à constante evolução dos ativos e serviços de TI, muitos casos tornam-se desatualizados ou não coerentes com a atual infraestrutura gerenciada.

Com o objetivo de desenvolver uma solução para a identificação de causas raiz de problemas, no presente artigo, apresentamos uma solução conceitual, apoiada por uma extensão do modelo *Common Information Model* (CIM) para a representação de informações sobre diagnóstico de problemas em infraestruturas de TI, que permite a reutilização total ou parcial dos casos.

Em contraste com as soluções existentes, nossa solução proposta é flexível em relação à constante evolução da infraestrutura de TI e adaptável para considerar novos problemas que possuam características similares à outros conhecidos. O processo de identificação de causa raiz auxiliado por esta solução permite a reutilização do conhecimento de casos nos diagnósticos que já foram completados e da experiência dos operadores.

Além disso, a solução proposta permite a identificação da causa raiz dos problemas analisados de uma forma dinâmica e interativa, considerando tanto a infraestrutura afetada, quanto as respostas produzidas pelo operador. Um estudo de caso, considerando falhas recorrentes em uma mudança de TI, é conduzido afim de comparar com a solução anterior proposta em [Santos *et al.* 2011], essa nova abordagem otimiza o processo de identificação de causa raiz, pois diminui a quantidade de interações entre o humano e o sistema, convergindo mais rapidamente para a resposta correta, bem como o impacto dos problemas para a continuidade dos negócios.

O restante do artigo está organizado conforme segue. Na Seção 2 são discutidos alguns dos principais esforços de pesquisas relacionados à erros e falhas em Gerenciamento de TI e áreas afins. Na Seção 3 é apresentada a solução para diagnóstico de problemas de

TI proposta no escopo deste artigo. Um estudo de caso conduzido para avaliar a eficácia da solução proposta é descrito na Seção 4, enquanto que na Seção 5 é concluído o artigo com considerações finais e perspectivas para trabalhos futuros.

2. Trabalhos Relacionados

A área de Gerenciamento de Mudanças em TI tem recebido grande atenção da comunidade científica nos últimos anos. As melhorias aplicadas aos processos de mudança os tornam mais eficientes, ágeis e reduzem seus custos. Diversos aspectos têm sido investigados, dentre eles avaliação de riscos associados às mudanças [Sauve *et al.* 2008], automação [Brown e Keller 2006], alinhamento a propósitos de negócio [Moura *et al.* 2008] e geração de planos de mudança de TI [Keller *et al.* 2004]. A seguir são discutidos alguns dos principais trabalhos relacionados a falhas em mudanças.

Machado *et al.* [Machado *et al.* 2008] propuseram uma solução para gerar planos de *rollback* antes da execução de uma mudança. Tal solução gera um plano de *rollback* para cada atividade atômica (reversível) da mudança, ou seja, é gerado um plano que desfaz as alterações realizadas durante a execução de um CP. Quando o *Deployment System* (sistema que realiza todas as alterações especificadas em um CP) detecta alguma falha, o plano de *rollback*, associado à atividade onde ocorreu a falha, é executado. Tal plano desfaz as mudanças executadas e retorna a infraestrutura gerenciada à um estado consistente. Porém, a causa da falha não é identificada. Além disso, a solução não reutiliza o conhecimento adquirido, permitindo que em outra execução da atividade, a falha possa ocorrer novamente, mas sem nenhum tratamento diferenciado.

O Gerenciamento de Problemas é a disciplina responsável por gerir o ciclo de vida de falhas, também chamadas de problemas e/ou incidentes [ITIL 2007]. São apresentados alguns conceitos, relacionamentos e fluxos, os quais auxiliam no diagnóstico das falhas. Porém, todas estas informações são tratadas em um alto nível de abstração. Visando suprir algumas carências do ITIL, Jäntti e Eerola [Jantti e Eerola 2006] propuseram um modelo conceitual orientado a negócios para o gerenciamento de problemas com suporte a defeitos e problemas de *software*. Tal modelo possibilita a documentação de todo o ciclo de vida de problemas dentro das organizações. Além disso, permite que o gerenciamento de problemas seja realizado tanto de forma pró-ativa, quanto reativa. Porém, não são abordados métodos e relacionamentos necessários para a identificação da causa raiz de falhas, realizando apenas a simples documentação do resultado final do diagnóstico.

Atualmente, os processos de tratamento de incidentes e problemas, são implementados de forma manual pelas organizações. Para resolver este problema, Gupta *et al.* [Gupta *et al.* 2008] propuseram uma solução para a automação do fluxo de processos do gerenciamento de incidentes, utilizando técnicas de integração de informações e aprendizagem de máquinas. Dentre essas técnicas está um método de correlação entre incidentes e Itens de Configuração (*CIs - Configuration Items*). Tal correlação possibilita o diagnóstico dos componentes que apresentam falhas quando incidentes são reportados. Baseado em informações de um dicionário de palavras, são indexadas palavras-chave para incidentes e seus relacionamentos. Comparando tais palavras-chave, são então identificados: os incidentes semelhantes ou recorrentes, o possível componente que falhou e as soluções utilizadas no passado. Tal solução possibilita com base em informações anteriores identificar o CI que falhou. Porém, apenas diagnosticar o CI que apresenta a falha não é o suficiente, pois este pode ter falhado por diferentes causas raiz.

Apesar da área de gerenciamento de problemas e incidentes tenha recebido grande atenção da comunidade científica em investigações recentes, nenhum dos trabalhos citados permite que seja identificada a verdadeira causa raiz para os incidentes reportados, ou em falhas ocorridas na execução de mudanças. Além disso, poucos trabalhos exploram a possibilidade de aprendizagem a partir de experiências passadas. Para tratar estas deficiências, neste artigo propomos uma solução conceitual, juntamente com uma expansão

do modelo *Common Information Model* (CIM), permitindo o suporte à identificação de causa raiz de falhas em execuções de Planos de Mudança (CP - *Change Plan*).

Ao analisarmos outras áreas relacionadas à falhas, podemos observar grandes avanços. Appleby *et al.* [Appleby *et al.* 2001] propuseram o *Yemanja*, um mecanismo baseado na correlação de eventos para diagnosticar falhas multi-camadas. Tal trabalho objetiva diagnósticos em cenários complexos de propagação de falhas e pode facilmente correlacionar eventos de redes de baixo nível, com alertas de aplicativos de alto nível, relacionando-os com violações na qualidade de serviços. Steinder e Sethi [Steinder e Sethi 2004] propuseram um método para diagnósticos de falhas em serviços *end-to-end*, baseado na análise de sintomas mapeados em grafos de causalidade e suas dependências. A solução permite o diagnóstico tanto de problemas de disponibilidade quanto de desempenho, em várias camadas da pilha de protocolos. Porém tais pesquisas objetivam encontrar a causa da falha de maneira autônoma. No entanto, nossa pesquisa está situada na área de gerenciamento de mudanças de TI, onde diferentes fontes de falhas são consideradas que não possibilitam diagnosticá-las de forma automática, como por exemplo, erros humanos.

3. Solução Proposta

Na solução proposta, a identificação da causa raiz (RC - *Root Cause*) de problemas é baseada em informações sobre diagnósticos de falhas anteriores, coletadas do próprio ambiente de TI. Em nossa proposta, tais informações foram obtidas a partir do ChangeLedge [Cordeiro *et al.* 2009], um sistema de gerenciamento de TI desenvolvido anteriormente pelo nosso grupo de pesquisa. Baseado nos dados obtidos junto ao ChangeLedge e com o atual estado da infraestrutura gerenciada, a solução proposta é capaz de adaptar-se ao histórico da falha e ao Item de Configuração (CI - *Configuration Item*) onde tal falha ocorre. Tal propriedade permite a dinamicidade do resultado do diagnóstico, pois apesar da falha e o CI identificados à priori serem os mesmos, a solução adapta-se as nuances da infraestrutura alvo bem como as respostas selecionadas pelo operador.

Neste trabalho, assumimos que as falhas ocorridas durante a implantação de mudanças são recorrentes, isto é, ao observar o histórico de execuções de um CP e as informações de diagnóstico associadas a este, é possível analisar falhas passadas e identificar as RCs mais prováveis. Assumimos também que as demais etapas do processo de gerenciamento de problemas serão tratadas por um sistema ou operador, focando na identificação das RCs e não na determinação da melhor solução para o problema.

Nesta seção será apresentada a solução para identificação interativa de RC de problemas/incidentes no gerenciamento de mudanças, considerando dois fatores que possibilitam a dinamicidade das perguntas a serem selecionadas: (*i*) o atual estado da infraestrutura gerenciada, que torna possível a seleção apenas de perguntas relevantes aos CIs envolvidos e (*ii*) os históricos de diagnósticos anteriores, que possibilitam a identificação das perguntas mais utilizadas. Em um primeiro momento, descreveremos o modelo que permite a representação das informações necessárias. Posteriormente, apresentaremos a arquitetura conceitual da solução. Por fim, introduziremos o processo de identificação, juntamente com os algoritmos que possibilitam o diagnóstico interativo.

3.1. Modelo de Informação

De acordo com o ITIL, as organizações devem manter um banco de dados que represente o atual estado de sua infraestrutura de TI (*e.g.*, Configuration Management Database - CMDB). Alguns modelos existentes contemplam boa parte das informações necessárias para descrever precisamente humanos, *software*, *hardware* e demais CIs. O *Common Information Model* (CIM) proposto pelo *Distributed Management Task Force* (DMTF) [DMTF 2009] é amplamente utilizado para este propósito. Esse modelo permite descrever de forma detalhada os itens de uma infra-estrutura de TI, tais como, sistemas

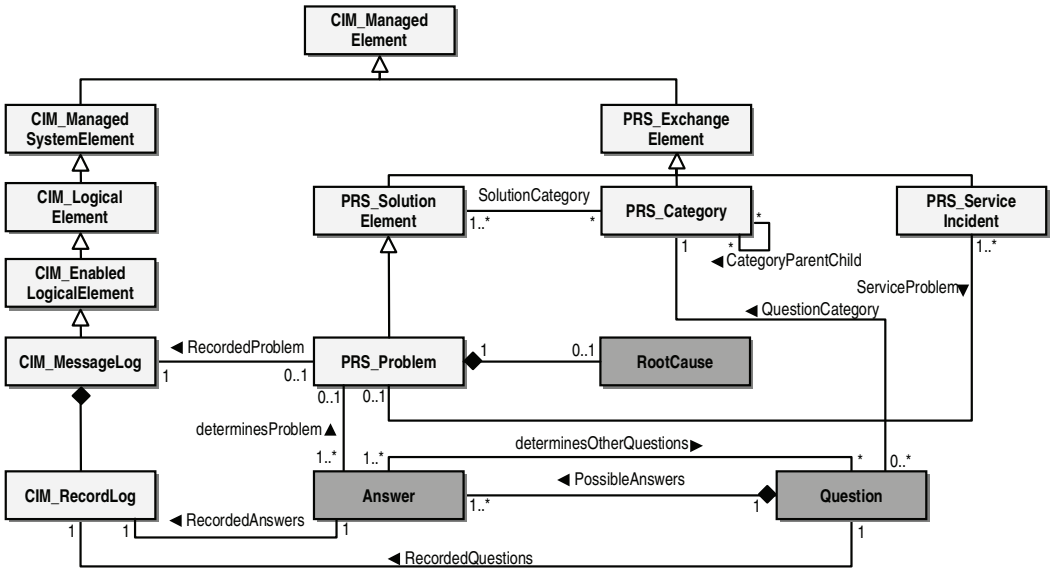

Figura 1. Extensão do modelo CIM para representar informações de diagnóstico

computacionais, pessoas ou processos e também as relações entre eles. Um conjunto específico de classes do CIM é destinado a representar informações sobre suporte, incluindo documentação de problemas e incidentes. Tais classes são referenciadas pelo DMTF como uma extensão, e são chamadas de *Problem Resolution Standard* (PRS). No entanto, esse modelo não fornece classes para a representação de RCs. Isto não permite que o conhecimento obtido com diagnósticos anteriores seja reaproveitado. Neste sentido, se faz necessário uma extensão do modelo CIM para que seja possível associar aos CIs informações sobre diagnóstico e reparo.

Para implementar a solução proposta e oferecer a um sistema o suporte a identificação de RC, é necessário estender o modelo CIM/PRS. Nossa expansão do modelo consiste basicamente na inclusão das classes: *Question*, *Answer* e *RootCause*. Na Figura 1 é apresentado uma simplificação do modelo proposto, onde as classes estendidas são apresentadas em cinza escuro e as classes do modelo CIM/PRS em cinza claro.

Uma *RootCause* é agregada à classe do *CIM_PRS_Problem*. Tal agregação possibilita a vinculação de tal causa tanto a um problema, quanto a vários incidentes. Um problema ou um incidente pode ou não possuir uma *RootCause* agregada e consequentemente um incidente pode ou não possuir uma *RootCause*.

Uma *Question* é uma agregação de possíveis *Answers*. Além disso, ela está associada a uma *PRS_Category*, isto permite identificar o nível de uma *Question*, ou seja, se uma pergunta é relacionada a um determinado tipo de dispositivo (ex.: "*Network Devices*"), ou se é de aspecto mais geral (ex.: "*Devices*"). Tal propriedade permite que perguntas de nível mais alto sejam selecionadas primeiro. De acordo com o ITIL, uma categoria pode possuir até três níveis de categorias pais/filhas (quatro níveis de categorias no total). Por exemplo, o CI "*Apache*" pertence as categorias "*Software*" e "*Web Server*". Além disso, a categoria "*Web Server*" é dependente de uma categoria de nível mais alto, neste caso, "*Software*". Portanto, nós consideramos a categoria "*Web Server*" pertencente ao Nível 2 e a categoria "*Software*" pertencente ao Nível 1.

Uma determinada *Answer* para uma *Question* pode ou não possibilitar a seleção de novas *Questions*. Um conjunto de determinadas *Answers* pode ou não determinar um *PRS_Problem* e, consequentemente, sua *RootCause*. Tanto a pergunta selecionada pelo *Diagnosis System* quanto a resposta selecionada pelo operador são registradas em um *RecordLog*. Uma agregação de *RecordLogs* é feita em um *MessageLog*, que contém todas as perguntas e respostas selecionadas durante um diagnóstico, bem como a RC identificada.

3.2. Arquitetura Conceitual

Baseado em uma arquitetura genérica de Gerenciamento de Mudanças, o Sistema de Diagnóstico (*Diagnosis System* - DS) foi introduzido para oferecer suporte à identificação da causa raiz de falhas. Na Figura 2 é apresentada a visão geral desta arquitetura conceitual, destacando os seus principais componentes, atores envolvidos e interações entre esses elementos. Os componentes, bem como o sistema, introduzidos neste artigo aparecem em destaque com o fundo cinza.

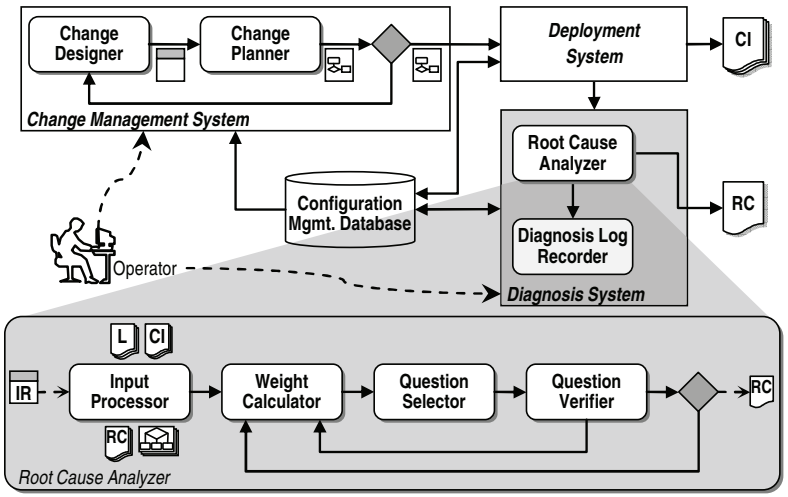

Figura 2. Arquitetura Conceitual da Solução Proposta

Um *Operator* interage com o *Change Management System* para criar uma*Request For Change* (RFC). A especificação inicial da RFC é realizada mais precisamente no componente *Change Designer*. Em linhais gerais, uma RFC descreve quais modificações devem ser acomodadas na infraestrutura gerenciada, os CIs afetados (*e.g.*, *firewall*, *switches*, serviços e aplicações) e os propósitos de negócio a serem alcançados. Os detalhes de implementação da mudança são esboçados pelo *Operator* na especificação de um Plano de Mudança (*Change Plan* - CP) preliminar, junto ao componente *Change Designer*. O plano preliminar consiste de um *workflow* de atividades (ou ações) que descrevem em um alto nível de abstração como a mudança requisitada deve ser materializada na infraestrutura.

O *Change Plan* preliminar é então refinado no componente chamado *Change Planner*. O resultado deste refinamento é um *workflow* com atividades de baixo nível que podem ser efetivamente executadas sobre os CIs. Tal *workflow* de atividades é chamado de CP refinado [Cordeiro *et al.* 2008]. Neste ponto, outras soluções podem ser aplicadas, *e.g.*, planos de *rollback* [Machado *et al.* 2008], gerência de riscos relacionada às mudanças [Sauve *et al.* 2008], análises de risco [Wickboldt *et al.* 2009] e/ou alocação de humanos [Lunardi *et al.* 2009] não detalhadas neste artigo. Logo após, o CP é analisado pelo *Operator* que decide por permitir a execução, ou retornar o CP ao componente *Change Designer* para possíveis alterações ou ajustes.

Assumindo que o *Operator* tenha aprovado o CP, este é executado pelo *Deployment System*. Tal sistema basicamente executa as mudanças descritas no CP sobre a infraestrutura de TI. Ao mesmo tempo, informações relevantes sobre a mudança que está sendo executada são armazenas em *logs*. Para cada atividade do CP que é concluída, o *Deployment System* faz a atualização no *Configuration Management DataBase* (CMDB) a fim de manter sempre uma visão atualizada da infraestrutura gerenciada.

Em um sistema com suporte a identificação de Causa Raiz (*Root Cause* - RC), uma falha é detectada pelo *Deployment System*, então é reportada ao *Operator* que pode

decidir entre identificar a RC ou executar outra solução. Ao optar pela identificação da RC, o *Operator* interage com o *Diagnosis System*, respondendo perguntas com as opções disponíveis. Essa seleção de perguntas é realizada pelo componente *Root Cause Analyzer* que será detalhada na próxima subseção. As informações que permitem a identificação da RC (*e.g.*, as perguntas, as respostas e as próprias RCs) estão armazenadas no CMDB. Após determinar a possível RC, o *Root Cause Analyzer* a informa ao *Operator*, que verifica se a identificação foi concluída com sucesso. No caso de uma resposta negativa, o *Operator* pode retornar ao processo de identificação, para realizar um novo diagnóstico.

Ao concluir o processo de identificação de RC, as informações relevantes sobre o diagnóstico realizado pelo *Operator* (*e.g.*, as perguntas selecionadas e respondidas e a RC identificada) são armazenadas em *logs* do sistema pelo componente *Diagnosis Log Recorder*, realimentando o sistema. Tais informações, além de serem reaproveitadas pelo *Root Cause Analyzer*, podem ser úteis para outros módulos ou soluções.

3.3. Root Cause Analyzer

Para auxiliar os operadores na identificação da RC, nós optamos por uma abordagem interativa. Basicamente o *Diagnosis System* o sistema seleciona uma pergunta que deve ser respondida pelo *Operator*. Tal interação pode gerar dinamicamente um novo *workflow* de diagnóstico a cada execução, baseado nas respostas do *Operator*. Optamos por tal abordagem pois ela permite que o sistema adapte a seleção de perguntas durante a execução à quatro fatores: ao problema/incidente que está sendo diagnosticado; à infraestrutura afetada; ao histórico de execuções; e às respostas informadas pelo *Operator*.

Assumindo que uma falha ocorreu durante a execução de um CP, o *Operator* deve criar um *Incident Report* (IR). O IR contém informações relevantes para o processo de identificação da RC, *e.g.*, o CI afetado pela falha, a prioridade e a o responsável e a identificação do IR. Tal relato pode ser realizado manualmente pelo *Operator* ou automaticamente pelo sistema. No último caso, o *Operator* apenas necessita decidir por identificar a RC da falha detectada pelo *Deployment System*.

O *Root Cause Analyzer* é o componente do *Diagnosis System* responsável pela identificação da RC. Como pode ser observado na Figura 2, para uma melhor compreensão, tal componente é dividido em quatro módulos: o *Input Processor*, o *Weight Calculator*, o *Question Selector* e o *Question Verifier*.

3.3.1. Input Processor

Baseado no CI informado no IR, o módulo *Input Processor* identifica quais os CIs dependentes, do CI previamente informado no IR. Isto irá listar todos os elementos que podem ter causado ou mesmo influenciado na falha. Tais dependências são mapeadas pelo CIM. Após identificar todos os elementos da infraestrutura que podem ter influenciado na falha, o módulo busca pelas categorias às quais os CIs pertencem. Assumindo que a categoria identificada possua múltiplos níveis, então todas as categorias superiores serão selecionadas.

Cada RC está associada a um conjunto que contém respostas específicas para determinadas perguntas. A identificação de uma RC ocorre quando tal conjunto é satisfeito em sua totalidade. Portanto, ao selecionar uma RC, as perguntas e respostas que identificam a RC também são selecionadas.

Tanto os CIs quanto as RCs estão associados às categorias. Além disso, dada uma categoria, é possível listar toda a infraestrutura que pode ter causado a falha. Também é possível identificar todas as RCs com base no CI informado pelo *Operator* no IR. Após ter listado todas as possíveis RCs para a falha ocorrida, os *logs* de diagnósticos passados são identificados. Ao finalizar um diagnóstico a RC identificada é gravada no *log*. Através

deste atributo é possível listar os diagnósticos anteriores em que uma determinada RC foi corretamente identificada. Tal atributo permite que o histórico possa ser consultado e considerado em novos diagnósticos.

3.3.2. Weight Calculator

Para possuir uma melhor possibilidade de identificar a RC correta, as RCs que possuem uma maior número de diagnósticos corretos são priorizadas. Essas informações são obtidas através dos *logs* do *Diagnosis System*. Os pesos atribuídos as categorias, perguntas e respostas são calculados com base no peso das RCs associadas a estes.

O peso de uma RC é calculado pelo soma dos diagnósticos concluídos em que a RC foi corretamente identificada. Porém o peso da RC recebe uma penalização para cada diagnóstico frustrado. Um diagnóstico é considerado frustrado, quando o DS identifica uma determinada RC, porém o *Operator* informa que este diagnóstico foi incorreto. Essa penalização consiste no somatório destes diagnósticos subtraindo do peso calculado previamente. Assumindo que uma RC foi identificada corretamente como a causa da falha em onze diagnósticos diferentes e tem cinco diagnósticos frustrados então o peso desta RC em um novo diagnóstico é igual a seis.

Algorithm 1 Weight Calculus

Require: S: Set of possible root causes, questions, answers, and categories associated to CIs identified; Log: logs of previous diagnoses.

1: **for all** $RC \in$ set of root causes from S **do**
2: $Weight_RC \leftarrow$WEIGHT_CALCULATE(RC,Logs)
3: **for all** $Q \in$ set of question from RC **do**
4: $Weight_Q \leftarrow$WEIGHT_ADD(Q,Weight_RC)
5: **for all** A associated with $RC \in$ set of answers from Q **do**
6: $Weight_A \leftarrow$WEIGHT_ADD(A,Weight_RC)
7: **end for**
8: **end for**
9: **for all** $C \in$ set of categories from RC **do**
10: $Weight_C \leftarrow$WEIGHT_ADD(C,Weight_RC)
11: **end for**
12: **end for**

Como apresentado no Algoritmo 1, o peso das categorias, perguntas e respostas é calculado com base no peso das RCs associadas. Para cada RC selecionada é atribuído o peso com base nos *logs* (Linha 2). Cada uma das perguntas (Linha 4), respostas (Linha 6), e categorias associadas (Linha 10) recebem o mesmo peso, somando com o atual peso destas. Essa adição possibilita uma pergunta ou categoria que está associada a várias RCs ter um peso maior do que uma que esteja associada a apenas uma dessas RCs. No entanto, a resposta tem o peso da RC adicionada apenas se estiver presente no conjunto de perguntas e respostas desta RC.

3.3.3. Question Selector

Após a atribuição dos pesos para todas as perguntas, respostas e categorias identificadas anteriormente, o módulo Question Selector seleciona uma pergunta para ser feita ao operador. Esta seleção é realizada com base nos pesos. Primeiramente é selecionada a categoria que possui o maior peso. Dentro desta categoria, a pergunta com o maior peso é

selecionada. Quando existir mais de uma pergunta com o mesmo peso, a pergunta que for de maior nível será selecionada, ou seja, a pergunta que não depender de nenhuma outra ou que for associada a categoria de nível mais alto.

3.3.4. Question Verifier

Baseado no histórico de execuções é possível identificar quantas vezes uma pergunta foi selecionada pelo sistema e respondida pelo operador. Esse histórico é mapeado nos pesos das perguntas e respostas. Após ser selecionada, uma pergunta pode ser considerada com óbvia ou não. Se considerada óbvia, a pergunta não é respondida pelo operador. A resposta que possuir o maior peso é automaticamente assumida pelo *Diagnosis System*. Caso a pergunta não seja considerada óbvia é solicitada a intervenção do operador, que deve selecionar entre uma das respostas disponíveis.

A obviedade de uma pergunta pode ser verificada pela análise dos pesos de suas respostas. Os passos para tal verificação são apresentados no Algoritmo 2. A primeira condição que deve ser satisfeita é o peso ser maior ou igual a um determinado *threshold* (Linha 2). Neste trabalho consideramos o peso 10 como *threshold*. É importante ressaltar que derivamos tal valor de experimentos realizados previamente em um ambiente simulado. De acordo com a necessidade de cada organização este valor pode ser ajustado, afim de adaptar a solução a diferentes ambientes. Este *threshold* permite que RCs com poucas execuções não sejam consideradas óbvias. Após, o módulo *Question Verifier* verifica se alguma das respostas disponíveis possui um peso maior ou igual a 80% do peso da pergunta (Linhas 3 e 4). Esta porcentagem é proposta pela técnica chamada de Análise de Pareto. Esta técnica é apresentada no ITIL e permite diferenciar causas potenciais das triviais [ITIL 2007]. Quando ambas as condições são satisfeitas a resposta óbvia é assumida pelo *Question Verifier* (Linha 5). No entanto, se qualquer uma das condições não for satisfeita, é solicitada a intervenção do operador, que deve responder a pergunta optando por um das respostas possíveis (Linhas 9 e 10).

Algorithm 2 Verification of Question

Require: Question selected, Answers associated, and their Weights calculated.
Ensure: R: Answer for the Question.

```
1:  R ← null
2:  if Weight_P >= threshold then
3:      for all A ∈ set of answers from P do
4:          if Weight_A >= 80% of Weight_P then
5:              R ← A
6:          end if
7:      end for
8:  end if
9:  if R is null then
10:     R ← OPERATOR_INTERVATION(P)
11: end ifreturn R
```

Supondo então que a pergunta "*The network card is configured?*" tenha um peso 23, e que a resposta "*Yes*" tem peso 19. Então tal pergunta não será enviada ao *Operator*, pois ela possui mais de 10 execuções e que em 82,60% dos casos a resposta foi a mesma. O *Question Verifier* assumirá tal resposta como se o *Operator* a tivesse informado.

Com base na resposta selecionada, o *Root Cause Analyzer* faz um refinamento no conjunto de possíveis RCs. As RCs que não podem ser mais identificadas são desconsideradas, bem como as perguntas associadas a estas. Os pesos das perguntas, respostas e categorias são recalculados com as RCs do conjunto após o refinamento. A seguir, os demais módulos são executados.

Após concluir o diagnóstico uma determinada RC é identificada. Tal RC é informada ao *Operator* que informará ao *Diagnosis System* se o diagnóstico foi concluído com sucesso. Caso a resposta seja afirmativa, o processo de identificação é encerrado e as perguntas, respostas selecionadas, as informações sobre o incidente e a RC identificada são armazenadas em *log*. Caso contrário, o processo é reiniciado, porém ignorando o módulo *Question Verifier*. Isto é realizado para evitar que seja atribuída uma resposta padrão a uma pergunta, que neste diagnóstico pode ser diferente.

4. Estudo de Caso

Para avaliar a viabilidade técnica da solução proposta, diferentes cenários foram analisados. Nesta seção, o estudo de caso apresentado é focado em uma falha ocorrida na execução de uma RFC instanciada em duas infraestruturas de TI distintas. Como resultados dessa análise nós esperamos observar a eficácia (checar se em ambos os casos as RCs são encontradas corretamente), a redução na quantidade de interações do humano com o sistema (checar se os *workflows* de diagnósticos resultantes convergem mais rapidamente para a correta identificação da RC) e a dinamicidade da solução (checar se os *workflows* de diagnóstico resultantes se adaptam tanto (*i*) a infraestrutura onde a falha ocorre quanto (*ii*) de acordo com as respostas informadas).

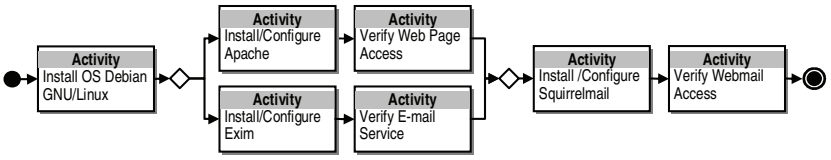

Figura 3. *Change Plan* para a instalação de um serviço de *webmail*

Na Figura 3 observam-se as principais atividades – em alto nível de abstração – envolvidas na RFC executada. Essa RFC tem o propósito de instalar um serviço de *webmail* (provido pelo *software* Squirrelmail) bem como suas dependências. Por exemplo, antes de instalar o Squirrelmail é necessário que exista instalado um serviço de páginas *web* (Apache) e um serviço de troca de e-mails (Exim). Em um primeiro cenário as atividades da RFC são instaladas em apenas um servidor, como apresentado na Figura 4(a). Por outro lado, em segundo cenário, as atividades relacionadas ao *software* de troca de e-mails (Exim) são instaladas em um servidor (E-mailServer) diferente de onde ficam hospedadas as páginas *web* (WebServer), como apresentado na Figura 4(b). Portanto, as infraestruturas disponíveis para a execução da mesma RFC são diferentes, possibilitando a ocorrências de falhas distintas. Em ambos os casos nós consideramos que a atividade "*Verify Webmail Access*" falhou.

De acordo com o *Diagnosis System* apresentado na Figura 2, um *Incident Report* (IR) é criado quando a falha na atividade "*Verify Webmail Access*" é detectada. Mais informações sobre o IR são fornecidas na Subseção 3.3. No próximo passo, o módulo *Input Processor* identificará os demais CIs dependentes relacionados com a atividade que falhou (*Verify Webmail Access*). Na Figura 4(a) e na Figura 4(b), podem ser observadas algumas dependências entre os CIs. "*Service Dependency*" representa a dependência incondicional para o funcionamento correto de serviços; "*Service Implementation*" representa a dependência que um serviço tem perante um *software*; "*InstalledSwElement*" representa a dependência da instalação de um *software* em um computador; e "*Hosted Service*" representa a dependência de um serviço por um *hardware*. Tais dependências e outras são mapeadas a partir do modelo CIM.

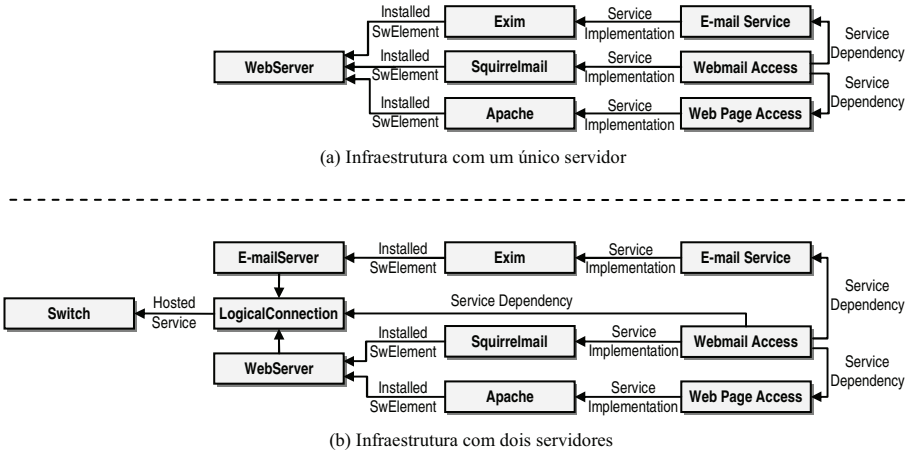

Figura 4. Instanciação da infraestrutura para a avaliação

Como apresentado na Subseção 3.3.1, após as dependências serem identificadas, o módulo *Input Processor* seleciona as categorias de cada CI. Com base nessas categorias são identificadas as possíveis RCs, perguntas, respostas e *logs* de diagnósticos anteriores. Logo após, o módulo *Weight Calculator* realiza o cálculo do peso das RCs, considerando diagnósticos corretos e os frustrados como descrito na Subseção 3.3.2. São então calculados, com base nos pesos das RCs, os pesos para as categorias, perguntas e respostas. Na Tabela 1 são representados os CIs e as categorias identificadas com os seus pesos calculados para o Cenário 1. Além disso, na Tabela 2, as mesmas informações são apresentadas para o Cenário 2.

Tabela 1. CIs, categorias com seus pesos associados para o Cenário 1

CI	Categorias	Pesos
E-mail Service	Service ⇒ E-mail	25⇒17
Web Page Access	Service ⇒ Web Page Server	25⇒7
Webmail Access	Service ⇒ Webmail	25⇒1
Exim	Software ⇒ Mail Server	35⇒15
Squirrelmail	Software ⇒ Webmail	35⇒9
Apache	Software ⇒ Web Server	35⇒1
WebServer	System ⇒ Computer System ⇒ Web Server	21⇒16⇒7

Como mencionado anteriormente na Subseção 3.3.3, o módulo *Question Selector* seleciona as perguntas para serem respondidas pelo *Operator*, respeitando os pesos de cada categoria. No Cenário 1, a ordem das categorias de Nível 1 é: *Software*, *Service* e *System*. Para o Cenário 2, a ordem das categorias de Nível 1 é: *Devices*, *Network*, *Software*, *System* e *Service*. É importante notar que os pesos são recalculados a cada interação. Além disso, o segundo cenário, por ter um número maior de CIs, tem listada um grande número de categorias, perguntas, respostas e RCs.

Por exemplo, o CI "*E-mail Service*" pertence a uma categoria de Nível 2, onde a sua categoria de Nível 1 é denominada "*Service*" e possui peso 25 e a de Nível 2 "*E-mail*" com peso 17. É importante mencionar que quando uma categoria é analisada, seu peso é o resultado da soma dos pesos das RCs pertencentes, incluindo o peso das RCs pertencentes as categorias dos níveis subsequentes, como apresentado anteriormente no Algoritmo 1.

Logo após, o módulo *Question Selector* seleciona a pergunta com base nos critérios definidos anteriormente (pesos), e o módulo *Question Verifier* verifica a obviedade da pergunta selecionada. Baseado na resposta do *Operator*, o conjunto de possíveis RCs é refinado (RCs e perguntas são descartadas desde que não possam mais ser identificadas). Este processo é repetido até que seja identificada uma RC. Esta RC é então

Tabela 2. CIs, categorias com seus pesos associados para o Cenário 2

CI	Categoria	Pesos
E-mail Service	Service ⇒ E-mail	25⇒17
Web Page Access	Service ⇒ Web Page Server	25⇒7
Webmail Access	Service ⇒ Webmail	25⇒1
Exim	Software ⇒ Mail Server	35⇒15
Squirrelmail	Software ⇒ Webmail	35⇒9
Apache	Software ⇒ Web Server	35⇒1
WebServer	System ⇒ Computer System ⇒ Web Server	26⇒21⇒5
E-mail Server	System ⇒ Computer System ⇒ Mail Server	26⇒21⇒7
Logical Connection	Network	38
Switch	Devices ⇒ Network Devices	40⇒36

informada ao *Operator* que determina o sucesso do diagnóstico.

Os *workflows* de diagnóstico resultantes para os Cenários 1 e 2 são exibidos nas Figuras 5(a) e 5(b) respectivamente. As perguntas exibidas com o fundo cinza, foram respondidas automaticamente pelo sistema. Isto ocorreu pois o *Question Verifier* considerou estas respostas como óbvias, ou seja, a intervenção do *Operator* não foi solicitada. As informações sobre o peso (quantidade de execuções para cada pergunta) e a porcentagem da mesma resposta são apresentadas abaixo de cada pergunta. Além disso, a linha "*Operator's Answer*" representa a resposta dada pelo *Operator*.

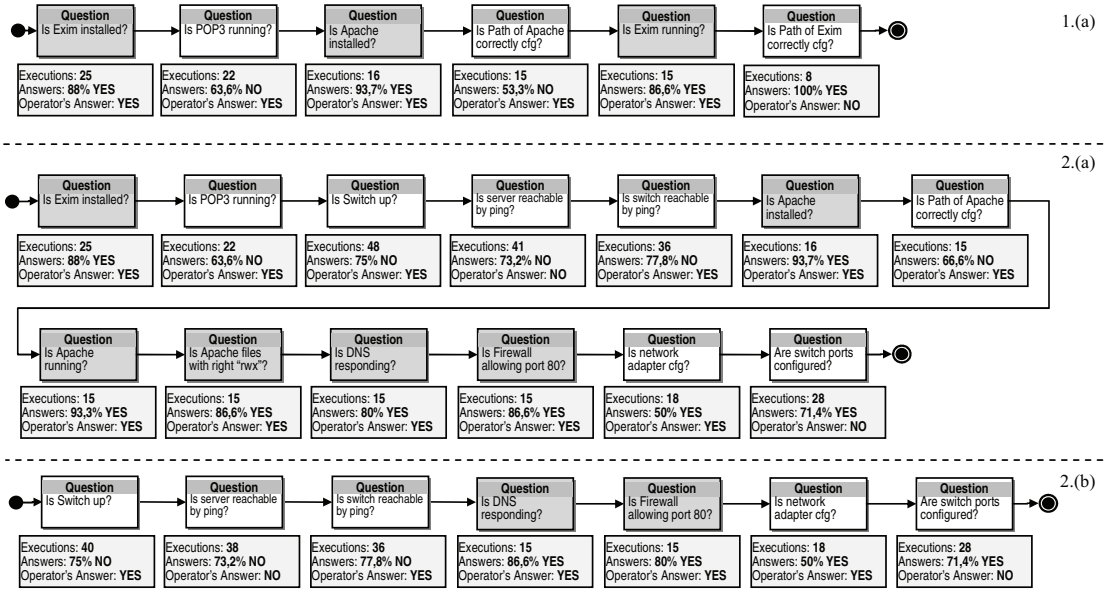

Figura 5. Workflow de Diagnósticos gerados pela solução

Cada resposta informada pelo *Operator* influencia diretamente na seleção da próxima pergunta. Isto resulta em uma sequência de diagnóstico dinâmica e ao mesmo tempo interativa, pois o *workflow* de diagnóstico é totalmente construído somente no término do processo de diagnóstico. Após o *Operator* terminar de responder a todas as perguntas em ambos os cenários, o *Diagnosis System* identificou ambas as RCs. Para o Cenário 1 foi identificada a RC "*Path to e-mail files is wrong*" e para o Cenário 2 foi identificada a RC "*The ports of the Switch were not configured properly*".

É importante enfatizar que essas duas RCs diferentes – encontradas em diferentes cenários – foram identificadas a partir da falha na mesma atividade. Além disso, é importante notar que os *workflows* gerados para cada cenário são diferentes. Adicionalmente, é

importante destacar que nossa solução não é capaz de identificar uma RC que não esteja documentada no CMDB, ou seja, uma RC que nunca ocorreu ou que ainda não foi identificada. Quando uma falha inédita ocorrer, ela somente será identificada pelo *Diagnosis System* após a sua documentação por um operador.

Nas Figuras 5.1.(a) e 5.2.(b) são apresentados os *workflows* gerados pela solução proposta para os Cenários 1 e 2 respectivamente. Na Figura 5.2(a) é apresentado o *workflow* de diagnóstico gerado para o Cenário 2, porém com a solução descrita em [Santos *et al.* 2011]. O *workflow* de diagnóstico do Cenário 1 foi omitido por ser equivalente em ambas as soluções. Como pode ser observado, os resultados produzidos pela versão aprimorada da solução, em cenários mais complexos, possuem uma menor quantidade de interações com o operador para identificar a RC. Tal melhoria foi concebida através do novo cálculo de pesos descrito neste artigo. A redução média observada na quantidade de interações foi de 20%.

5. Conclusões e Trabalhos Futuros

O processo de identificação de causa raiz representa uma etapa fundamental para a gerência e a operação de infraestruturas de TI. No entanto, as soluções existentes para auxiliar essa etapa não utilizam métodos que possam reaproveitar facilmente o conhecimento adquirido. Consequentemente, falhas na execução de mudanças podem ocorrer indefinidamente sem que os operadores saibam exatamente o que está causando as mesmas. Para abordar esse problema, neste artigo é proposta uma solução para a identificação de RCs de falhas em mudanças de TI utilizando questões e respostas dinamicamente. Esta abordagem torna o processo de diagnóstico interativo e dinâmico.

Os resultados obtidos durante o estudo de caso conduzido, mostram os benefícios de utilizar nossa solução proposta para a identificar as causas raiz de falhas nas atividades de mudanças de TI. Conforme apresentado anteriormente, a solução leva em consideração as nuances do ambiente e se adapta às variações da infraestrutura de TI. A partir de uma mesma falha, a solução gerou diferentes *workflows* de diagnóstico, considerando os CIs envolvidos, o histórico de execuções anteriores e as respostadas informadas pelo operador. Por fim, é realizada uma comparação com uma solução proposta anteriormente, a qual demonstra uma melhoria nos diagnósticos realizados, visto que o sistema encontra a causa raiz realizando um número menor de interações com o operador.

Como trabalhos futuros, são previstas algumas melhorias na atual versão de nossa solução. Entre elas destacam-se quatro: (*i*) investigar estratégias para aperfeiçoar o módulo *Question Selector*, considerando outras métricas, *e.g.*, a idade dos diagnósticos na seleção de perguntas, a fim de priorizar causas raiz recentes; (*ii*) adotar heurísticas que permitam otimizar os *workflows* gerados para a identificação das causas raiz; (*iii*) investigar o uso das dependências para diagnosticar os requisitos das CIs para seu correto funcionamento, afim de melhorar o processo de *bootstrapping* do *Diagnosis System*; (*iv*) estender o processo de identificação de causas raiz para outros escopos, *e.g.*, aplicando os mesmos métodos para gerenciar incidentes em operações/suporte de serviços.

Referências

Appleby, K., Goldszmidt, G., e Steinder, M. (2001). Yemanja-a layered event correlation engine for multi-domain server farms. Em *Integrated Network Management Proceedings, 2001 IEEE/IFIP International Symposium on*.

Brown, A. e Keller, A. (2006). A best practice approach for automating it management processes. Em *Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP*, páginas 33 –44.

Cordeiro, W., Machado, G., Andreis, F., *et al.* (2009). ChangeLedge: Change design and planning in networked systems based on reuse of knowledge and automation. *Computer Networks*, 53(16):2782 – 2799.

Cordeiro, W. L. C., Machado, G., Daitx, F., *et al.* (2008). A template-based solution to support knowledge reuse in IT change design. Em *Network Operations and Management Symposium, 2008. NOMS 2008. IEEE*, páginas 355–362.

DMTF (2009). Distributed Management Task Force: Common Information Model. Distributed Management Task Force (DMTF). Available: http://www.dmtf.org/standards/cim. Accessed: nov. 2009.

Gupta, R., Prasad, K., e Mohania, M. (2008). Automating itsm incident management process. Em *Autonomic Computing, 2008. ICAC '08. International Conference on*, páginas 141 –150.

ITIL (2007). ITIL - Information Technology Infrastructure Library: Service Operation Version 3.0. Office of Government Commerce (OGC).

ITIL (2010). ITIL - Information Technology Infrastructure Library. Office of Government Commerce (OGC). Disponível em: http://www.itil-officialsite.com/. Acessado em: out. 2010.

Jantti, M. e Eerola, A. (2006). A Conceptual Model of IT Service Problem Managementz. Em *Service Systems and Service Management, 2006 International Conference on*, volume 1, páginas 798–803.

Keller, A., Hellerstein, J., Wolf, J., Wu, K.-L., e Krishnan, V. (2004). The champs system: change management with planning and scheduling. Em *Network Operations and Management Symposium, 2004. NOMS 2004. IEEE/IFIP*, páginas 395–408.

Lunardi, R. C., Wickboldt, J. A., Cordeiro, W. L., *et al.* (2009). ChangeAdvisor: Alinhando o Planejamento de Mudanças em Infra-estruturas de TI a Objetivos/Restrições de Negócios. Em *XXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2009)*, páginas 437–450.

Machado, G., Daitx, F., Cordeiro, W., *et al.* (2008). Enabling rollback support in IT change management systems. Em *Network Operations and Management Symposium, 2008. NOMS 2008. IEEE*, páginas 347–354.

Moura, A., Sauve, J., e Bartolini, C. (2008). Business-driven it management - upping the ante of it : exploring the linkage between it and business to improve both it and business results. *Communications Magazine, IEEE*, 46(10):148 –153.

Santos, R., Wickboldt, J., R., L., Dalmazo, B., Granville, L., Gaspary, L., Bartolini, C., e Hickey, M. (2011). A Solution for Identifying the Root Cause of Problems in IT Change Management. Em *Proceedings of Mini-conference of 2011 IFIP/IEEE International Symposium on Integrated Network Management (IM 2011). To appear*.

Sauve, J., Santos, R., Reboucas, R., Moura, A., e Bartolini, C. (2008). Change priority determination in it service management based on risk exposure. *Network and Service Management, IEEE Transactions on*, 5(3):178 –187.

Sauvé, J. P., Santos, R. A., Almeida, R. R., *et al.* (2007). On the Risk Exposure and Priority Determination of Changes in IT Service Management. Em *XVIII IFIP/IEEE International Workshop on Distributed Systems: Operations and Management (DSOM 2007)*, páginas 147–158.

Steinder, M. e Sethi, A. S. (2004). Probabilistic fault diagnosis in communication systems through incremental hypothesis updating. *Computer Networks*, 45(4):537 – 562.

Wickboldt, J. A., Lunardi, R. C., Machado, G. S., *et al.* (2009). Automatizando a Estimativa de Riscos em Sistemas de Gerenciamento de Mudanças em TI. Em *XXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2009)*, páginas 423–436.