UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

RICARDO LUIS DOS SANTOS

Identificação Interativa da Causa Raiz de Problemas no Gerenciamento de Mudanças de TI

Dissertação apresentada como requisito parcial
para a obtenção do grau de
Mestre em Ciência da Computação

Prof. Dr. Lisandro Zambenedetti Granville
Orientador

Porto Alegre, Junho de 2012

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Santos, Ricardo Luis dos

Identificação Interativa da Causa Raiz de Problemas no Gerenciamento de Mudanças de TI / Ricardo Luis dos Santos. – Porto Alegre: PPGC da UFRGS, 2012.

88 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2012. Orientador: Lisandro Zambenedetti Granville.

1. Identificação de Causa Raiz. 2. Diagnóstico Interativo. 3. Gerenciamento de Problemas. 4. Gerenciamento de Mudanças. I. Granville, Lisandro Zambenedetti. II. Título.

“As oportunidades multiplicam-se
à medida em que são agarradas.”
— SUN TZU

AGRADECIMENTOS

Primeiramente, devo meus sinceros agradecimentos aos meus pais e irmão pelos ensinamentos, compreensão, conselhos e principalmente pelo apoio em todos os momentos. Vocês me ensinaram o verdadeiro significado de família, amizade, união, garra, comprometimento e cumplicidade. Quero que saibam, todas as conquistas em minha vida são resultados dos seus ensinamentos. Muito obrigado!

Meu agradecimento especial ao meu orientador Lisandro que sempre me ajudou quando precisei e, além disso, soube exigir quando necessário, um verdadeiro amigo. Estendo esse agradecimento aos professores Luciano Gaspary, Marinho Barcellos e Juergen Rochol que de alguma forma contribuíram para o meu crescimento profissional e/ou pessoal.

Ao "gurizedo" do grupo de redes, não apenas pelas diversas histórias inesquecíveis, mas pelas lições que aprendi com vocês: Juliano (Vingador), Rubão, Weverton, Fabrício, Alan (Pirulito), Alan (Mona), Nautilus ("ooo meu!"), Pietro (Píton), Felipe (Galinho), Fábio Daitx, Paulo (Priumo), Rafael (Iraquiano), Barata, Jeff, Kunst, Raniery, Cristiano, Abraham Lincoln, JEdi, Jaruba, Rodolfo, Adler ("comi merda bicho!"), Oscar (primo do Pablito), Catatau, Goiano, Matheus Gus (Mateus Lima), Esteves, Emannuel (Japa), Paolo, Marotta, Felipe (Escudero), Presunto O Grande, Angel, Matheus, Bondan, Purinho, Rodrigo (Goku), Leonardo (o cara parecido com o Rodolfo) e a todos os demais integrantes com quem tive o prazer de conviver. Em especial tenho que agradecer aos "irmãos paleta!" (Dalmazo e Mansilha) pelas festas, viagens e histórias protagonizadas, nossa amizade é para sempre. Paleta!

Por fim, mas não menos importante, agradeço a algumas pessoas que me ajudaram no decorrer dessa caminhada (pô, são 3 anos...). Vandão, Mauro Moreira, Tácio Castanhede e Caciara, sem o apoio de vocês eu não teria iniciado o mestrado e ficaria "tranquilo" no meu antigo emprego no Banrisul. Nilza, Edevir, Fernanda e Cacá, amigos que me acolheram quando cheguei em Porto Alegre. Agradeço também a todos os amigos que fiz e que me ajudaram no decorrer desses 3 anos, entre eles, Abraão, Bonga, Xixo, Jeca, Thais, Nani, Oyá, Carla, Larissa e Lidi. Em especial a Bruna, além do carinho, compreensão e companheirismo sempre evidentes, tu soubeste o exato momento de puxar minhas orelhas e de me apoiar quando necessário.

Muito obrigado a todos!

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

AMN	*Activity Modeling Notation*
BS	*British Standard*
CAB	*Change Advisor Board*
CCTA	*Central Communications and Telecommunications Agency*
CIs	*Configuration Items*
CIM	*Common Information Model*
CMDB	*Configuration Management Database*
CMS	*Change Management System*
CPs	*Change Plans*
COBIT	*Control Objectives for Information and related Technologies*
DML	*Definitive Media Library*
HP	*Hewlett-Packard*
HSBC	*Hongkong and Shanghai Banking Corporation*
IBM	*International Business Machines*
ISACA	*Information Systems Audit and Control Association*
ISO	*International Organization for Standardization*
IT	*Information Technology*
ITIL	*Information Technology Infrastructure Library*
ITSM	*Information Technology Service Management*
KEDB	*Known Error Database*
NASA	*National Aeronautics and Space Administration*
OGC	*Office of Government Commerce*
PMBOK	*Project Management Body of Knowledge*
PMI	*Project Management Institute*
PR	*Problem Record*
PRS	*Problem Resolution Standard*

RC	*Root Cause*
RFC	*Request For Change*
SLA	*Service Level Agreement*
TI	Tecnologia da Informação

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Atualmente, o ambiente extremamente dinâmico e altamente competitivo, no qual as organizações estão inseridas, aumentou expressivamente a importância da Tecnologia da Informação (TI). Aliada a isto, a crescente complexidade das infraestruturas de redes e serviços, também conhecidas como infraestruturas de TI, tornaram a Gerência de Serviços de Tecnologia da Informação indispensável a qualquer organização. Nesse cenário, coletâneas de boas práticas e processos foram desenvolvidas visando a obter o melhor desempenho dos produtos e serviços oferecidos. Entre as coletâneas amplamente aceitas, tanto na academia quanto na indústria, vale destacar a Biblioteca de Infraestrutura de Tecnologia da Informação (*i.e., Information Technology Infrastructure Library*, ITIL). Dentre os processos descritos na ITIL, o Gerenciamento de Mudanças é de suma importância para as organizações. Tal processo é responsável por padronizar o planejamento, o agendamento, a implementação e a avaliação de mudanças no ambiente de TI.

Ainda que a adoção das boas práticas proporcione significativas melhorias, a ocorrência de falhas em procedimentos de mudanças não pode ser negligenciada. Para tratar eventuais falhas é proposto o processo de Gerenciamento de Problemas, o qual é responsável por gerir o ciclo de vida de problemas. Nesse contexto, o reúso do conhecimento e da experiência do operador, sobre os processos de TI, é de fundamental importância, pois possibilita a simplificação de procedimentos para a detecção da causa raiz de falhas e, consequentemente, minimiza perdas financeiras 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 reutilizar o conhecimento em falhas recorrentes ou similares. Diante do exposto, nesta dissertação é proposta uma solução para a identificação interativa da causa raiz de problemas detectados em mudanças. Em contraste com as soluções existentes, o foco desta dissertação é o reúso de traços históricos de diagnósticos e da experiência do operador.

A solução proposta teve sua aplicabilidade avaliada em dois estudos de casos. Os cenários utilizados são baseados em situações reais, bem como consideram falhas recorrentes em mudanças sobre as infraestruturas de redes e serviços. Os resultados demonstram a capacidade da solução em reutilizar o conhecimento adquirido com experiências anteriores, bem como, na adaptabilidade do diagnóstico gerado, tanto na infraestrutura de TI na qual a falha ocorre, quanto nas respostas fornecidas pelo operador. A estrutura modular da solução desenvolvida permite dividir a complexidade do processo de identificação em problemas menores que podem ser resolvidos individualmente. Assim, as organizações podem personalizar partes da solução para melhor refletirem as necessidades específicas do seu ambiente de TI.

Palavras-chave: Identificação de Causa Raiz, Diagnóstico Interativo, Gerenciamento de Problemas, Gerenciamento de Mudanças.

Interactive Identification of Root Cause of Problems in IT Change Management

ABSTRACT

Presently, the extremely dynamic and highly competitive environment, in which organizations are inserted, significantly increased the importance of Information Technology (IT). In addition, the increasing complexity of network infrastructures and services, also known as IT infrastructures, turned the Information Technology Service Management essential to any organization. In this context, guides of best practices and processes were developed to obtain a best performance out of provided products and services. Among the most widely accepted proposals, in both academy and industry, it is worth mentioning the ITIL (Information Technology Infrastructure Library). From the processes described in ITIL, Change Management is indispensable for organizations. This process defines how changes should be planned, scheduled, implemented, and evaluated in IT environments.

Despite the significant improvements that the adoption of ITIL's best practices can provide, the occurrence of failures cannot be neglected. To address possible failures, the process of Problem Management is proposed. Such process is responsible for managing the life cycle of problems. Thus, the reuse of operator's knowledge and experience, on IT processes, has fundamental importance for allowing to simplify the procedures of detecting failures' root cause and, therefore, minimizes financial losses and maintenance costs. Nevertheless, the diagnostic process, in the existing solutions, is performed in a static and ad hoc fashion, which complicates the reuse of knowledge in recurrent failures or similar. Based on the previous considerations, in this dissertation, we propose a solution for interactive identification of the root cause of problems detected in IT changes. In contrast to existing solutions, the focus of this solution is the reuse of the operator's experience and the knowledge from cases that have already been completed.

The proposed solution had its applicability evaluated in two case studies. The scenarios used are based on real situations, taking into account recurrent failures in changes on IT infrastructures. The results show not only the ability of the solution to reuse the knowledge acquired from previous experiences, but also the adaptability of diagnosis generated in both the IT infrastructure, where failure occurs, and the responses provided by operator. The modular structure of the developed solution allows to divide the complexity of the identification process into smaller problems that can be solved individually. Thus, organizations can customize parts of the solution to better reflect the specific requirements of your IT environment.

Keywords: Root Cause Identification, Interactive Diagnostic, Problem Management, Change Management.

1 INTRODUÇÃO

Atualmente, o ambiente extremamente dinâmico e altamente competitivo, no qual as organizações estão inseridas, aumentou expressivamente a importância da Tecnologia da Informação (TI). Aliada a isto, a crescente complexidade das infraestruturas de redes e serviços, também conhecidas como infraestruturas de TI, tornou a Gerência de Serviços de Tecnologia da Informação (*i.e., Information Technology Service Management*, ITSM) indispensável a qualquer organização (SAUVÉ et al., 2007). Assim sendo, oferecer serviços de TI com alta disponibilidade, segurança e desempenho deixou de ser apenas mais um objetivo do departamento de tecnologia e tornou-se um dos principais alicerces para o sucesso das organizações.

Nesse cenário, coletâneas de boas práticas e processos foram desenvolvidas visando a alcançar uma melhor qualidade no fornecimento de serviços, bem como alinhar os custos dos processos às estratégias das organizações. Uma das mais importantes referências nesse contexto, a Biblioteca de Infraestrutura de Tecnologia da Informação (*i.e., Information Technology Infrastructure Library*, ITIL) (OGC, 2011), tem ajudado as organizações a manter de forma apropriada seus ativos e serviços de TI. Além disso, as boas práticas contidas na ITIL são de fundamental importância, em organizações que possuem serviços em larga escala e altamente dinâmicos, pois tornam os processos vinculados à ITSM mais ágeis e com custos reduzidos.

Apesar de serem extremamente comuns no ambiente corporativo, ações como adição, remoção e/ou atualização da infraestrutura de TI, por mais simples que sejam, necessitam de um gerenciamento de seu ciclo de vida. Isso porque, além de possuírem altos custos atrelados, tais ações são suscetíveis a diversas falhas, que podem resultar na redução da qualidade, ou em casos mais extremos, na interrupção dos serviços prestados pelas organizações. Com o propósito de mitigar a ocorrência dessas falhas, diversas diretrizes são propostas na ITIL para o processo do Gerenciamento de Mudanças (*i.e., Change Management*). Tal processo é responsável por padronizar os procedimentos que envolvam o planejamento, agendamento, implementação e/ou a avaliação de mudanças no ambiente de TI (OGC, 2007a).

Ainda que a adoção das boas práticas da ITIL proporcione significativas melhorias para a operação e gerência de infraestruturas de TI, a ocorrência de falhas em procedimentos de mudanças não pode ser negligenciada pelos operadores e tomadores de decisão. Para tratar eventuais falhas é proposto o processo de Gerenciamento de Problemas (*i.e., Problem Management*), o qual é responsável por gerir o ciclo de vida dos problemas de TI (OGC, 2007b). Os principais objetivos de tal processo são: (*i*) prevenir a ocorrência de problemas e incidentes resultantes em infraestruturas de TI; (*ii*) eliminar incidentes recorrentes; e (*iii*) minimizar o impacto de incidentes que não possam ser evitados. Para alcançar tais objetivos, o reúso do conhecimento e da experiência do operador, sobre os

processos de TI, é de fundamental importância, pois possibilita a simplificação de procedimentos para a detecção da causa raiz de falhas e, consequentemente, reduz os custos associados.

A fim de auxiliar no processo de diagnóstico, as organizações utilizam ferramentas de apoio à análise e identificação da causa raiz de problemas. Tais ferramentas contêm passos pré-definidos que ajudam a diagnosticar e mitigar falhas recorrentes. Porém, apesar de apresentarem alguma eficácia, a descrição de casos nessas ferramentas é realizada de forma estática e *ad hoc*, por exemplo, através do uso de *scripts* que contenham perguntas e respostas. Além disso, a reutilização do conhecimento mapeado nesses casos é dificultada devido à constante evolução dos ativos e serviços de TI. Da mesma forma, muitos casos tornam-se desatualizados ou não coerentes com a atual infraestrutura gerenciada. Isso limita a identificação da causa raiz em casos similares e, consequentemente, reduz a probabilidade de sucesso do diagnóstico.

Ambas as áreas, Gerenciamento de Mudanças e Gerenciamento de Problemas, têm recebido grande atenção da comunidade científica nos últimos anos. Diversas pesquisas foram desenvolvidas com o foco em mudanças e falhas, por exemplo, a categorização automática de problemas (DIAO; JAMJOOM; LOEWENSTERN, 2009), os planos de *rollback* (MACHADO et al., 2008), os riscos associados às mudanças (SAUVÉ et al., 2008) e a identificação de falhas de *softwares* (AGARWAL; MADDURI, 2010). Porém, os trabalhos mencionados não identificam a causa raiz de problemas e, em alguns casos, são soluções específicas para detecção de falhas em *softwares*. Além disso, os casos descritos nesses trabalhos não são flexíveis em relação às nuances do ambiente de TI, bem como não reutilizam o conhecimento que foi mapeado em diagnósticos de falhas anteriores.

Diante do exposto, nesta dissertação é proposta uma solução para a identificação da causa raiz de problemas detectados em mudanças de TI. Em contraste com as soluções existentes, o foco desta dissertação é o reúso de traços históricos de diagnósticos e da experiência do operador. Isso possibilita otimizar o processo de identificação da causa raiz de falhas em mudanças de TI e, consequentemente, reduzir o impacto, os custos atrelados e o tempo despendido nesse processo. De posse disso, na solução desenvolvida foram enumerados os requisitos listados a seguir.

1. Ser flexível às evoluções da infraestrutura de TI – como apresentado anteriormente, mudanças na infraestrutura de TI são extremamente constantes no ambiente corporativo. Nesse sentido, a solução deve considerar a infraestrutura que é afetada e/ou que pode ser a responsável pela falha relatada.

2. Possuir casos adaptáveis à falhas similares – a impossibilidade de reutilizar as descrições dos casos, em falhas com características similares, é uma das maiores limitações para a utilização de *scripts* pré-definidos no diagnóstico de falhas. Para resolver essa limitação, faz-se necessário desenvolver um modelo de representação de casos que possibilite o reaproveitamento total ou parcial dos mesmos.

3. Ser compatível com o padrão de representação de infraestruturas – um dos padrões para representação de infraestruturas de TI mais utilizados atualmente é o *Common Information Model* (CIM) (DMTF, 2008). Tal padrão permite a representação de informações sobre ativos e serviços de TI. A fim de permitir a compatibilidade com as soluções em uso pelas organizações, propõe-se adaptar o modelo CIM, permitindo a representação de informações sobre diagnóstico de problemas em infraestruturas de TI.

4. Reutilizar o conhecimento adquirido por experiências anteriores – segundo a ITIL, reutilizar o conhecimento adquirido é de suma importância, pois permite a redução de custos e torna os processos envolvidos mais ágeis. Logo, informações sobre diagnósticos anteriores devem ser consideradas, bem como a experiência adquirida ao longo do tempo pelos operadores.

5. Permitir ao operador interagir com o sistema – por estar inserida no Gerenciamento de Problemas reativo, a solução deve considerar informações submetidas pelo operador. Tal característica permite à solução adaptar-se às respostas informadas, considerando tanto o histórico obtido do próprio ambiente de TI quanto a experiência dos operadores.

A fim de provar os conceitos da solução proposta nesta dissertação, dois estudos de caso são conduzidos. Ambos utilizam cenários baseados em casos reais, bem como consideram falhas recorrentes em mudanças nas infraestruturas de TI. No primeiro cenário são descritas duas infraestruturas diferentes: a primeira considera um ambiente instalado em apenas um servidor; a segunda infraestrutura descreve os mesmos elementos, porém instalados em diferentes servidores. Apesar da simplicidade do primeiro cenário, o principal objetivo é demonstrar a adaptabilidade da solução às nuances da infraestrutura de TI. O segundo cenário, descreve a infraestrutura de uma empresa com atuação na área de serviços Web, por exemplo, hospedagem de *sites*, *e-mail* e comércio eletrônico. O principal objetivo com esse estudo de caso é demonstrar o correto funcionamento da solução considerando um ambiente corporativo real, destacando as demais características da solução, entre elas, a flexibilidade, a interação, o reúso do conhecimento e a compatibilidade com os padrões atualmente utilizados.

O restante desta dissertação está organizado conforme segue. No Capítulo 2 são apresentados os principais conceitos e discutidos alguns dos principais esforços relacionados, tanto ao Gerenciamento de Mudanças quanto ao Gerenciamento de Problemas. No Capítulo 3 é apresentada a solução conceitual para diagnóstico de problemas na execução de mudanças de TI. Além disso, são discutidas quatro estratégias para a seleção de uma pergunta em cada interação, bem como as suas peculiaridades. Dois estudos de caso são conduzidos para avaliar a solução proposta no Capítulo 4, enquanto que no Capítulo 5 é concluída a dissertação com as considerações finais e as perspectivas para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA E TRABALHOS RELACIONADOS

A área de ITSM tem recebido massiva atenção nos últimos anos. Isto se deve à crescente importância dos recursos e serviços de TI para a continuidade dos negócios das organizações (SAUVÉ et al., 2007). Assim sendo, coletâneas de boas práticas e processos como o *Control Objectives for Information and related Technologies* (COBIT) mantido pela *Information Systems Audit and Control Association* (ISACA) (ISACA, 2011), a ITIL de autoria da *Office of Government Commerce* (OGC) (OGC, 2011) e o *Project Management Body of Knowledge* (PMBOK) publicado pela *Project Management Institute* (PMI) (PMI, 2011), auxiliam as organizações a manterem, de forma apropriada, seus ativos e serviços de TI. Por possuir uma maior adoção mundial, nesta dissertação, as investigações foram focadas na ITIL.

O capítulo está organizado conforme segue. As principais fases do ciclo de vida de um serviço de TI, bem como um breve histórico sobre a ITIL são apresentados na Seção 2.1. O fluxo do processo de Gerenciamento de Mudanças, que possibilita planejar, elaborar, agendar e executar mudanças na infraestrutura de TI é descrito na Seção 2.2. A fim de descrever as atividades que permitem gerenciar o ciclo de vida de problemas, os principais conceitos e atividades do processo de Gerenciamento de Problemas são apresentados na Seção 2.3. Na Seção 2.4, são discutidas as principais pesquisas relacionadas ao assunto desta dissertação.

Desenvolvido pelo grupo de Redes de Computadores da Universidade Federal do Rio Grande do Sul, o sistema CHANGELEDGE visa a automatizar o planejamento e a execução de mudanças sobre as infraestruturas de TI. Tal sistema aglomera componentes desenvolvidos em diferentes pesquisas, entre eles, a estimativa de riscos e a alocação de humanos. A compreensão do funcionamento desse sistema é de fundamental importância, pois permite introduzir a solução para identificação da causa de uma falha em um sistema real e aceito pela comunidade científica. Tal sistema é apresentado na Seção 2.5.

2.1 ITIL *Core* v3

A ITIL é a coletânea de boas práticas e processos para gerência de serviços de TI mais adotada mundialmente. Com autoria da *Central Communications and Telecommunications Agency* (CCTA), que posteriormente foi integrada ao OGC, sua primeira versão foi publicada no Reino Unido pela *Her Majesty's Stationery Office*, entre os anos de 1989 a 1995. Os princípios descritos foram baseados em histórias de sucesso de diversas organizações, além de diferentes pesquisas realizadas por consultores, especialistas e doutores (OGC, 2011).

No ano de 2005, os princípios da ITIL e o padrão britânico BS 15000, editado pela *British Standards Institution*, serviram como base para a criação da norma ISO 20000. Essa norma é a primeira, editada pela *International Organization for Standardization* (ISO), que versa sobre a gestão de serviços de TI (ISO, 2005). Atualmente, inúmeras empresas públicas e privadas estão adequadas à ISO 20000 e, consequentemente à ITIL, por exemplo, Toyota, Disney, NASA, IBM, HP e HSBC.

A terceira versão da ITIL, publicada em 2007, consiste em dois componentes: o *ITIL Complementary Guidance* e o ITIL *Core* V3. O *ITIL Complementary Guidance* é um conjunto de publicações que contém modelos de operação, arquiteturas tecnológicas e orientações específicas para organizações industriais. O *ITIL Core* V3 consiste em cinco publicações que representam as diferentes fases do ciclo de vida de um serviço de TI, que são: *Service Strategy*; *Service Design*; *Service Transition*; *Service Operation*; e *Continual Service Improvement*. Tais publicações descrevem os principais processos e boas práticas aplicáveis a todas as organizações. É importante salientar que os conceitos descritos na ITIL e que são referenciados nesta dissertação, seguem a nomenclatura retirada do glossário oficial em português (OGC, 2011). As fases do ciclo de vida de um serviço de TI, bem como os seus principais processos[1] são apresentados na Figura 2.1 e, brevemente descritos a seguir.

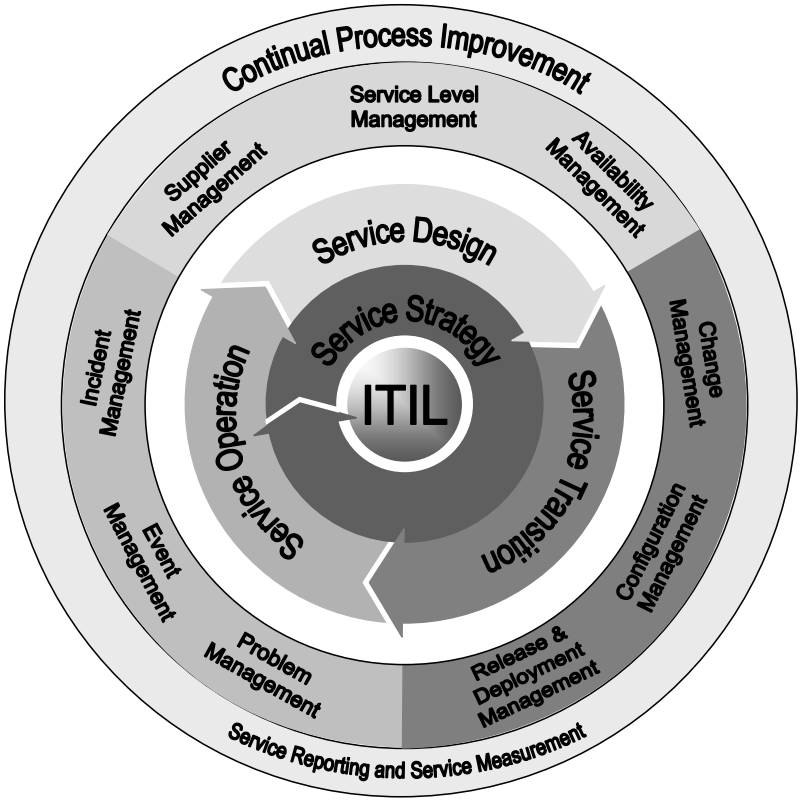


Figura 2.1: As fases do ciclo de vida de serviços e os principais processos segundo a ITIL Core V3

[1]Alguns processos integram mais de uma fase do ciclo de vida do serviço de TI, portanto, são citados em mais de uma publicação. Por exemplo, o Gerenciamento de Problemas é descrito na fase de Operação de Serviço (*i.e., Service Operation*), na qual são definidos o objetivo, as técnicas, as atividades e métodos. Por outro lado, na fase de Melhoria Contínua de Serviço (*i.e., Continual Service Improvement*), o Gerenciamento de Problemas é novamente citado, porém, como forma para avaliar e melhorar os serviços prestados aos clientes.

Inicialmente, a fase de *Estratégia de Serviço* (*i.e., Service Strategy*) fornece as principais diretrizes sobre como projetar, desenvolver e implementar a ITSM, não apenas como uma função administrativa interna, mas como um ativo estratégico, ou seja, a ITSM deve ter um desempenho notável, consolidando-se como uma vantagem competitiva perante os concorrentes. As organizações utilizam tais diretrizes para identificar, selecionar e priorizar oportunidades mercadológicas, facilitando a tomada de decisões, visando a realizar bons investimentos, bem como estabelecer novos objetivos para os serviços prestados. Os principais processos recomendados nessa publicação são o Gerenciamento da Demanda, o Gerenciamento Financeiro e o Gerenciamento de Portfólio de Serviços.

A segunda fase do ciclo de vida recebe o nome de *Desenho de Serviço* (*i.e., Service Design*) e fornece orientações para o correto planejamento e desenvolvimento dos serviços de TI. Essa publicação abrange princípios e métodos que visam a converter objetivos estratégicos em serviços. O escopo dessa fase não se limita apenas a novos serviços, mas abrange as mudanças e melhorias necessárias para manter, ou até mesmo incrementar, o valor dos serviços existentes. Além disso, nessa fase são projetadas tecnologias para atender às necessidades dos clientes a um baixo custo, considerando diversas métricas, por exemplo, o conhecimento da equipe, os Acordos de Níveis de Serviço (*i.e., Service Level Agreement*, SLA) e a eficiência. Entre os processos descritos nessa publicação estão o Gerenciamento de Fornecedores, o Gerenciamento dos Níveis de Serviço e o Gerenciamento da Disponibilidade.

A terceira fase, *Transição de Serviço* (*i.e., Service Transition*), busca garantir que os serviços novos, modificados ou obsoletos, sejam entregues, como documentado nas fases anteriores. Caso exista a necessidade de alterações na especificação do serviço, estas serão realizadas durante a fase de transição, a fim de que se entregue o serviço de acordo com o requerido pelo cliente. Ademais, essa fase foca em todos os aspectos da implementação do serviço de TI, devendo garantir tanto o correto funcionamento, em circunstâncias extremas ou anormais, quanto o suporte à falhas ou erros, que porventura ocorram. Os principais processos dessa fase são o Gerenciamento de Mudanças, o Gerenciamento de Liberação e Implantação e o Gerenciamento de Configurações.

Na visão dos consumidores, é somente durante a fase de *Operação de Serviço* (*i.e., Service Operation*) que os serviços agregam valor ao negócio, pois é somente nesse momento que os serviços desenvolvidos são colocados em operação. Tal fase visa a entregar aos clientes e usuários os serviços contratados dentro dos limites dos SLAs previamente acordados. É importante observar que diversos objetivos dessa fase são conflitantes, necessitando que os gestores tomem decisões importantes, visando a manter ou a incrementar o valor agregado aos serviços. Tais decisões caracterizam um *trade-off*, em que maximizar o ganho em um determinado objetivo acarreta em detrimento de outro. Esse *trade-off* pode ser constatado através da análise de métricas, tais como, qualidade e custo. Um serviço de TI entregue com uma alta qualidade, normalmente resulta em um maior custo para a organização. Os principais processos dessa fase são o Gerenciamento de Incidentes, o Gerenciamento de Eventos e o Gerenciamento de Problemas.

Por fim, a fase de *Melhoria Contínua de Serviço* (*i.e., Continual Service Improvement*) busca avaliar e melhorar a qualidade dos serviços, bem como a melhoria geral da ITSM e seus processos, em três diferentes níveis. O primeiro nível é a execução integrada de todos os processos da ITSM. O segundo é o alinhamento contínuo do portfólio de serviços de TI às necessidades atuais e futuras do negócio. O último nível, trata da maturidade requerida do serviço de TI para suportar os processos do negócio. Além disso, essa fase visa a aperfeiçoar a qualidade, a eficiência e a eficácia dos processos, reduzindo os custos

efetivos na entrega de serviços, respeitando os SLAs contratados. Entre os principais processos dessa fase estão a Medição do Serviço e o Relatório de Serviço.

É importante salientar que os processos descritos na ITIL não formam uma metodologia, mas sim um conjunto das melhores práticas na área de ITSM. Nessas publicações, não é definido como os processos devem ser executados, mas apenas o que, para que e por que eles devem ser desenvolvidos dentro das organizações.

2.2 Gerenciamento de Mudanças

O Gerenciamento de Mudanças é o processo responsável por controlar as mudanças na infraestrutura de TI, com o objetivo de otimizar os recursos envolvidos, gerar o menor impacto nos serviços providos e assegurar que, ao término das mudanças, a infraestrutura de TI permaneça em um estado consistente. Nesse contexto, o termo mudança compreende qualquer alteração nos Itens de Configuração (*i.e., Configuration Items, CIs*), por exemplo, *switches*, *firewall*, roteadores, serviços e aplicações. Dessa forma, estão inclusos desde pequenos ajustes, por exemplo, reiniciar um dispositivo ou dar permissões de acesso, até a implementação de novos serviços ou soluções conforme as necessidades da organização.

É importante salientar que é recomendada a utilização de três componentes distintos pelas organizações para auxiliar no processo de Gerenciamento de Mudanças. O primeiro, o Sistema de Gerenciamento de Configurações (*i.e., Configuration Management System*) é responsável por coletar, armazenar, gerenciar, atualizar, analisar e apresentar dados sobre todos os CIs e seus relacionamentos. O Banco de Dados de Gerenciamento de Configurações (*i.e., Configuration Management Database*, CMDB) armazena os dados e relacionamentos de todos os CIs envolvidos nos processos da ITSM. Além disso, o CMDB é utilizado como fonte para consulta de dados por componentes que necessitem de informações sobre os CIs. Por fim, a Biblioteca de Mídia Definitiva (*i.e., Definitive Media Library*, DML) armazena todas as versões definitivas e autorizadas de *softwares* utilizados pela organização.

A ITIL descreve as atividades do processo de Gerenciamento de Mudanças conforme apresentado na Figura 2.2. Primeiramente, um documento denominado Requisição de Mudança (*i.e., Request For Change*, RFC) é criado (*Create RFC*). Tal documento é uma solicitação formal feita pelo requisitante da mudança. Esse requisitante pode ser um departamento dentro da organização ou um indivíduo, por exemplo, um cliente. Caso a mudança solicitada tenha um impacto razoável na continuidade dos negócios ou significantes implicações financeiras, um documento denominado Proposta de Mudança (*Change Proposal*) pode ser necessário. Esse documento inclui informações relevantes sobre os serviços introduzidos e que servirão de base para a análise e aprovação da mudança.

Posteriormente, a mudança solicitada é armazenada no CMDB (*Record the RFC*). Diversas informações devem ser armazenadas no momento da criação do registro de uma RFC, por exemplo, a descrição, a justificativa, a prioridade e a categoria. Além disso, tal registro é constantemente atualizado, por exemplo, ao armazenar as atividades que foram executadas, necessárias para a conclusão da RFC solicitada. O grau de detalhe dessas informações depende do impacto e do tamanho da mudança analisada.

Logo após, as RFCs são avaliadas em relação à integridade e à viabilidade técnica e financeira (*Review RFC*). As mudanças que não cumpram com os padrões exigidos ou que forem consideradas impraticáveis, devem ser retornadas ao requisitante da mudança, juntamente com a justificativa para tal. Caso seja aprovada, a mudança é novamente

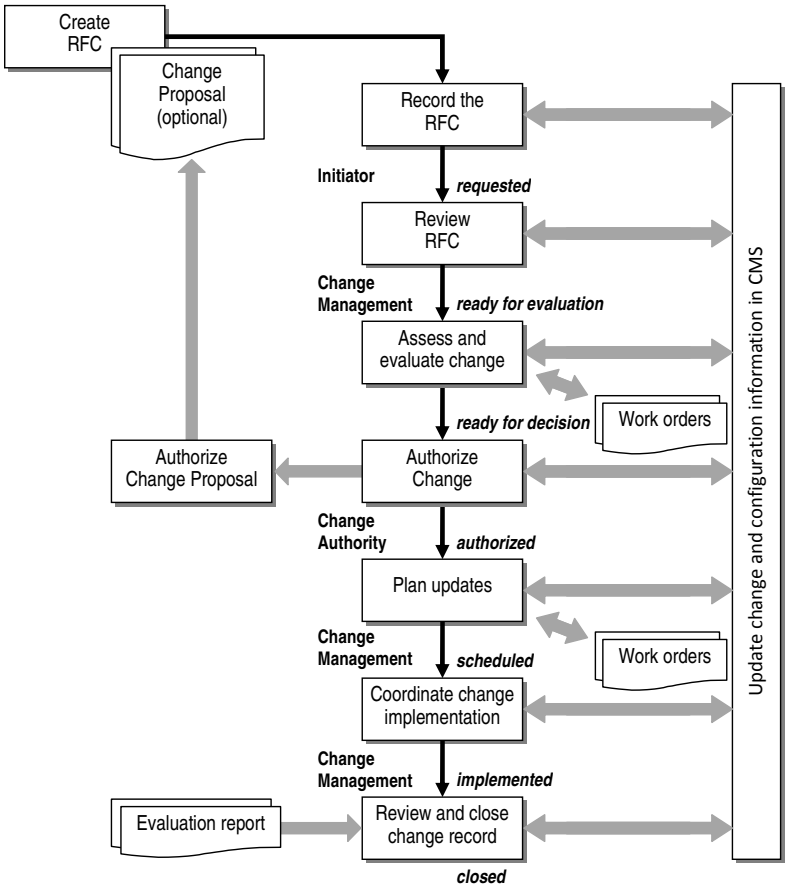


Figura 2.2: O fluxo do processo de mudanças segundo a ITIL Core V3

avaliada, porém, desta vez pelo Comitê Consultivo de Mudanças (*i.e., Change Advisor Board*, CAB). O CAB é um grupo de pessoas capazes de analisar e avaliar as mudanças, tanto do ponto de vista técnico, quanto do ponto de vista empresarial (*Assess and evaluate change*). Caso a RFC seja autorizada (*Authorize Change*), as ações necessárias para realizá-la são planejadas, atualizadas e testadas (*Plan updates*). Posteriormente, a RFC é encaminhada para os responsáveis pela implantação sobre a infraestrutura de TI (*Co-ordinate Change Implementation*). A ITIL recomenda a utilização de documentos denominados *Work orders*, que facilitam o rastreamento de atividades em execução. Tais documentos são requisições formais para a execução de uma determinada atividade.

Finalmente, após a conclusão da mudança, os resultados são submetidos para avaliação pelos responsáveis (*Review and close change record*). É nessa atividade que incidentes observados durante a implantação da mudança devem ser relatados. Tal atividade é de suma importância para que incidentes recorrentes possam ser mais facilmente detectados e solucionados. Além disso, nessa etapa os requisitantes da mudança, bem como os clientes afetados, devem ser informados, a fim de verificarem se a mudança atingiu o resultado esperado.

2.3 Gerenciamento de Problemas

A ITIL (OGC, 2007b) define um problema como a causa desconhecida para um ou mais incidentes. Um incidente é definido como a interrupção de um serviço ou a redução

na sua qualidade, por exemplo, a indisponibilidade do servidor de *e-mail* ou a redução na velocidade de navegação Web. Além disso, uma causa raiz é o fato causador de um problema que, ao ser corrigido, evita a recorrência de incidentes. Em um exemplo hipotético, após atualizar a tabela de roteamento de um dispositivo da rede, diversos computadores perderam a conectividade com a Internet. Nesse caso, o fato de não ser possível acessar a Internet é considerado um incidente para cada computador. O problema caracteriza-se pelo fato causador que originou tais incidentes ser desconhecido. Após a conclusão dos diagnósticos, determina-se que a causa raiz foi a alteração da rota *default* do roteador para um IP inválido.

O processo de Gerenciamento de Problemas é responsável por gerenciar o ciclo de vida de todos os problemas, e tem por objetivos: (*i*) evitar a ocorrência de problemas e incidentes nas infraestruturas de TI; (*ii*) eliminar incidentes recorrentes; e (*iii*) minimizar o impacto de incidentes que não possam ser evitados (OGC, 2007b). Tal processo é desdobrado em dois domínios: o domínio pró-ativo que busca evitar que problemas ou incidentes ocorram; e o domínio reativo que prioriza a resolução dos problemas relatados. Esse último é o foco desta dissertação. A ITIL descreve as principais atividades do processo de Gerenciamento de Problemas no domínio reativo conforme apresentado na Figura 2.3.

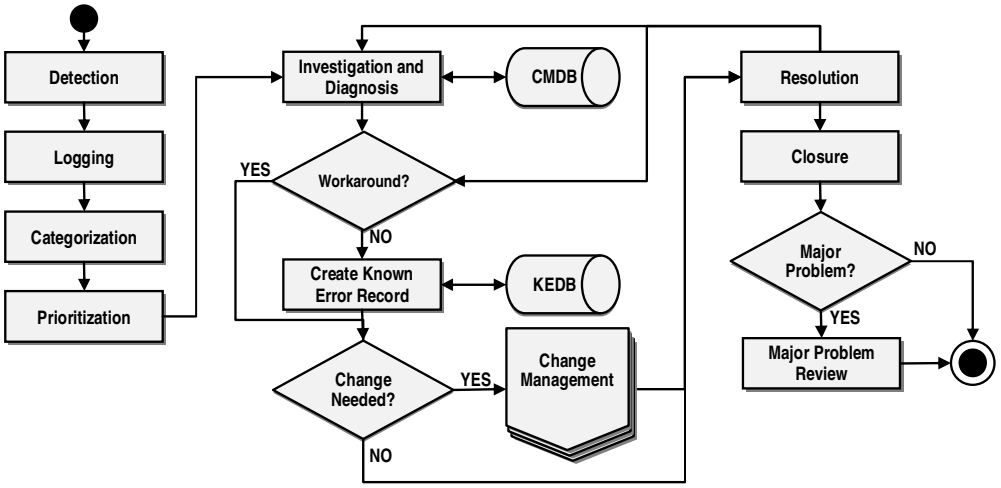


Figura 2.3: O fluxo do Gerenciamento de Problemas segundo a ITIL Core V3

Para que o Gerenciamento de Problemas possa ser iniciado, faz-se necessário que ocorra a detecção de uma falha (*Detection*). Tal detecção pode ser realizada de diferentes formas, por exemplo, através da análise de um incidente por um grupo de suporte, verificações automáticas, tendências observadas em relatórios, por notificações de um fornecedor ou através do *Service Desk*. Logo após, informações relevantes são obrigatoriamente registradas (*Logging*), por exemplo, o CI que está apresentando a falha, detalhes do serviço e do usuário, além de informações sobre a detecção e uma breve descrição da falha. Essas informações geram um documento denominado Registro de Problema (*i.e., Problem Record*, PR). Tal documento contém todos os detalhes iniciais e registra todo o ciclo de vida do problema que o originou.

Posteriormente, ocorre a categorização do problema (*Categorization*). Nessa atividade uma categoria é atribuída ao PR relatado. Essa atribuição é de fundamental importância, pois permite agrupar problemas semelhantes, auxiliando no diagnóstico e na

localização de uma solução. Para tanto, a categorização deve ser realizada utilizando categorias multi-níveis com no máximo quatro níveis de granularidade.

Na atividade seguinte, ocorre a definição da prioridade do PR relatado (*Prioritization*). A prioridade busca determinar o prazo para a resolução do problema. Segundo a ITIL, uma maneira simples para determinar a prioridade de um problema utiliza o impacto e a urgência. O impacto define qual o nível do serviço que está afetado. Por exemplo, em uma empresa de *e-commerce*, a queda do servidor Web afeta todos os serviços, portanto, o impacto desse problema é alto. A urgência refere-se ao tempo que a organização está disposta a esperar para a resolução do problema. No exemplo anterior, a urgência é alta, visto que sem o *site* a empresa deixa de faturar.

Para grandes organizações é recomendada a utilização de uma terceira métrica, a severidade. A severidade busca mensurar a seriedade do problema na perspectiva da infraestrutura de TI, considerando outras informações, por exemplo, o custo e os recursos humanos necessários para a resolução do problema. Essa métrica visa a melhorar a definição da prioridade de um PR. Porém, não são especificados quais níveis e como essa nova métrica é utilizada juntamente com as definidas anteriormente. Por outro lado, na Tabela 2.1 é apresentado um sistema simples para codificação da prioridade de um PR, enquanto que na Tabela 2.2 é apresentado um exemplo de prazos negociados nos SLAs.

Tabela 2.1: Sistema simples para codificação de prioridades

		Impacto		
		Alto	**Médio**	**Baixo**
Urgência	**Alto**	1	2	3
	Médio	2	3	4
	Baixo	3	4	5

Tabela 2.2: Tempo para resolução dos problemas relatados

Código da Prioridade	**Descrição**	**Tempo para Resolução**
1	Crítico	1 hora
2	Alto	8 horas
3	Médio	24 horas
4	Baixo	48 horas
5	Planejar	Planejado

De posse dessas informações, a atividade de investigação e diagnóstico é iniciada (*Investigation and Diagnosis*). Essa atividade busca determinar a causa raiz do problema relatado, ou seja, definir qual a real causa da falha detectada. Para tanto, são utilizadas diferentes técnicas, por exemplo, sessões de *brainstorming* entre os responsáveis, análise de pareto e análises cronológicas. Tais técnicas podem utilizar informações de diagnósticos anteriores, bem como da infraestrutura afetada, ambas disponíveis em um CMS.

Em alguns casos, após encontrar a causa raiz, é possível desenvolver uma solução de contorno (*Workaround*). Essa solução visa a reduzir ou a eliminar rapidamente o impacto, sem necessariamente resolver o incidente/problema, por exemplo, reiniciando um CI em falha. É importante salientar que mesmo existindo uma solução de contorno, ainda é necessário o desenvolvimento de uma resolução definitiva. Caso a falha detectada esteja ocorrendo pela primeira vez, ou seja, não exista nenhum histórico de ocorrência, é criado

um registro de erro conhecido, onde serão armazenadas informações relevantes sobre o diagnóstico, por exemplo, *status*, a causa raiz e em alguns casos a solução de contorno (*Create Known Error Record*). Esse registro será utilizado em futuros diagnósticos e deve ser criado em um Banco de Dados de Erros Conhecidos (*i.e., Known Error Database*, KEDB).

Uma solução de contorno pode exigir que alterações sejam realizadas nos CIs, então uma RFC é criada e encaminhada para o Gerenciamento de Mudanças. Após a conclusão da mudança requerida, uma atividade para resolver definitivamente a falha é inicializada (*Resolution*). Quando a resolução da falha detectada preliminarmente não elimina os incidentes, novos diagnósticos são necessários a fim de detectar a real causa da falha. Isto faz com que o processo retorne a atividade de investigação e diagnóstico. Caso a falha seja corrigida, bem como os incidentes relatados, a resolução é documentada e armazenada no KEDB.

Por fim, o encerramento formal do problema é realizado (*Closure*). Essa atividade limita-se em alterar apenas o *status* do PR para encerrado. Porém, caso o problema seja considerado grave, informações relevantes sobre as diferentes atividades devem ser armazenadas no KEDB (*Major Problem Review*). Tais informações visam a auxiliar futuros diagnósticos, bem como agilizar a resolução de problemas recorrentes minimizando o impacto nos negócios.

2.4 Trabalhos Relacionados

A grande área de ITSM tem recebido grande atenção da comunidade científica nos últimos anos. Diversos aspectos têm sido investigados, por exemplo, a avaliação de riscos associados às mudanças (SAUVÉ et al., 2008), a detecção de Planos de Mudança conflitantes (HAGEN; KEMPER, 2011), a reutilização do conhecimento (RAHMOUNI; BARTOLINI, 2010), a alocação de recursos (SHUAI; ZHIQIANG, 2011), o alinhamento a propósitos dos negócios (MOURA; SAUVE; BARTOLINI, 2008) e contratos eletrônicos (KELLER, 2005). A seguir serão discutidos alguns dos principais trabalhos, na área de ITSM, relacionados a mudanças e a falhas.

Em uma das primeiras pesquisas publicadas, na área de Gerenciamento de Mudanças, Keller *et al.* (KELLER et al., 2004) propuseram um sistema chamado CHAMPS, que automatiza a geração de Planos de Mudança (*i.e., Change Plans*, CPs). A fim de facilitar o uso e a implantação, esse sistema utiliza tecnologias bem difundidas, por exemplo, *workflows*. Além disso, o sistema busca explorar a paralelização na execução das atividades. Isso melhora o desempenho na execução dos CPs gerados. Por tratar-se de um dos primeiros esforços, a referida pesquisa não abrange a totalidade do processo de mudanças, não considerando aspectos importantes, por exemplo, a associação de recursos às atividades e a especificação de objetivos e/ou restrições na geração automática de CPs.

Em um trabalho posterior, Brown e Keller (BROWN; KELLER, 2006) propuseram uma abordagem, baseada nas boas práticas da ITIL, que visa a automatizar os processos envolvidos na ITSM. Esse trabalho aborda mudanças que envolvam especificamente o ciclo de vida de *softwares*, ou seja, que incluam atividades que manipulem *softwares*, tais como, instalação, atualização, remoção e ou configuração. A solução desenvolvida busca paralelizar atividades com o intuito de otimizar o processo de mudança. Essa paralelização resulta na execução de um CP mais rápido e com custo reduzido, pois as atividades são automatizadas, não sendo necessária a intervenção humana para a execução.

Visando à automatização, Cordeiro *et al.* (CORDEIRO et al., 2009) propuseram o

ChangeLedge, um sistema que possibilita gerar CPs detalhados com base em informações de alto nível. A solução permite que, na criação da RFC, tarefas rotineiras sejam especificadas em um alto nível de abstração e, posteriormente, refinadas pelo sistema em passos detalhados. Para elaborar um CP detalhado, o sistema utiliza planos preliminares, pacotes de *softwares* e suas dependências, bem como informações sobre a infraestrutura de TI. Além disso, são utilizados *snapshots* da infraestrutura, ou seja, representações de estados intermediários que podem ocorrer durante a execução do CP. Como resultado, os CPs gerados tendem a estar menos propensos a falhar, reduzindo assim a ocorrência de incidentes relacionados às execuções de atividades mal dimensionadas.

Em um trabalho posterior, Lunardi *et al.* (LUNARDI et al., 2010) propuseram o ChangeAdvisor, um sistema de suporte à decisão que, além de alinhar os CPs refinados aos propósitos de negócios, permite a alocação de recursos humanos às atividades. Para atingir tais objetivos, seis diferentes estratégias são descritas. Essas estratégias exploram diferentes formas de alocação de humanos e permitem aos administradores de TI compreenderem os *trade-offs* envolvidos. Através de uma melhor compreensão dos *trade-offs* é possível priorizar objetivos contrastantes, por exemplo, tempo de execução da mudança e o custo atrelado.

Na publicação *Service Operation* da ITIL *Core* V3 são apresentados alguns conceitos, relacionamentos e fluxos, os quais auxiliam no diagnóstico das falhas (OGC, 2007b). Porém, essas informações são tratadas em um alto nível de abstração. Com o objetivo de melhor detalhar as atividades envolvidas, Jäntti e Eerola (JANTTI; EEROLA, 2006) propuseram um modelo conceitual orientado a negócios para o Gerenciamento de Problemas. Tal modelo possibilita a documentação de todo o ciclo de vida de defeitos e problemas de *softwares* dentro das organizações. Além disso, permite que o Gerenciamento de Problemas seja realizado tanto no domínio pró-ativo, quanto reativo. 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.

Em busca de automatizar a categorização dos problemas relatados, Diao *et al.* (DIAO; JAMJOOM; LOEWENSTERN, 2009) propuseram uma solução colaborativa baseada em regras. Na solução proposta são definidos dois tipos de usuários: os autores, profissionais experientes que criam as regras utilizadas no sistema; e os consumidores, que utilizam tais regras para classificar os problemas relatados. Com o objetivo de simplificar e agilizar o processo de classificação, as regras são divididas em inclusivas e exclusivas. Primeiramente, para cada problema relatado as regras exclusivas são executadas. Tais regras quando possuem as condições satisfeitas excluem a categoria do conjunto de códigos válidos. Logo após, são executadas as regras inclusivas. Nesse caso, ao serem satisfeitas as condições da regra, o problema relatado é classificado com a categoria atrelada.

Em outra pesquisa, Bartolini e Stefanelli (BARTOLINI; STEFANELLI, 2011) abordam o alinhamento dos processos de TI aos objetivos de negócios, definidos nos *Service Levels Objectives*. Nessa pesquisa é descrito um sistema de apoio à decisão aplicável em diferentes domínios, tais como, gerenciamento de mudanças e gerenciamento de incidentes. Esse sistema consiste em componentes que podem ser reutilizados, além de diretrizes e padrões que permitem o desenvolvimento de soluções específicas. A aplicação dessas soluções permite reduzir os custos e incrementar a agilidade na execução dos processos. Ademais, é descrito um algoritmo para priorização de incidentes, que utiliza os objetivos de negócios para definir a prioridade de resolução de cada incidente relatado (BARTOLINI; SALLE; TRASTOUR, 2006).

A fim de minimizar o impacto das falhas que ocorrem durante o processo de mudan-

ças, Sauvé *et. al* (SAUVÉ et al., 2007) e Rebouças *et al.* (REBOUÇAS et al., 2007) propuseram soluções para a análise de riscos. As soluções desenvolvidas realizam a análise de riscos alinhada aos objetivos de negócio e priorizam automaticamente as RFCs quando são escalonadas para a execução. Isto permite executar, em momentos apropriados e com uma maior atenção, mudanças que envolvam funcionalidades sensíveis aos objetivos da organização. Nessas pesquisas, a análise de riscos é realizada na fase de escalonamento das mudanças a serem implantadas. Tal análise permite mitigar riscos entre a submissão da RFC e a sua execução. Porém, durante a execução de uma RFC, podem ocorrer falhas que ocasionam perdas financeiras ou quebras de SLAs, por exemplo, a interrupção de serviços e a violação de prazos.

Com o intuito de analisar riscos durante o planejamento de mudanças, Wickboldt *et al.* (WICKBOLDT et al., 2011) propuseram um *framework* para a análise compreensiva de riscos. Esse *framework* estima os riscos de cada uma das atividades descritas no CP. Tal estimativa é baseada em traços de execuções anteriores desse CP ou de CPs similares, isto é, CPs que contenham atividades idênticas ou similares. A fim de melhorar a precisão do cálculo da similaridade das atividades e, consequentemente, melhorar a estimativa de riscos, Bianchin *et. al.* (BIANCHIN et al., 2010) propuseram um algoritmo que permite calcular a similaridade das atividades de um CP com base em informações de execuções de CPs similares. A correta análise dos riscos calculados, na fase de planejamento, identifica as atividades que devem receber uma maior atenção, por exemplo, a configuração de um servidor DNS deve ser realizada por profissionais mais experientes. Esse melhor planejamento reduz a probabilidade de certas falhas ocorrerem. Porém, apenas reduzir a probabilidade não impede a ocorrência de falhas.

Como uma forma para tratar falhas ocorridas em mudanças, Machado *et al.* (MACHADO et al., 2008) propuseram uma solução baseada na geração de planos de *rollback*. A solução gera um plano de *rollback* associado a cada atividade reversível do CP, ou seja, é gerado um plano que desfaz as alterações realizadas durante a execução de um CP. Quando uma falha é detectada durante a execução de um CP, o plano de *rollback* associado à atividade que falhou é executado. Tal plano desfaz as mudanças executadas e retorna a infraestrutura gerenciada a 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.

A fim de reutilizar o conhecimento adquirido com falhas recorrentes, Gupta *et al.* (GUPTA; PRASAD; MOHANIA, 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 CIs. Tal correlação permite 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. Essa solução possibilita, com base em informações anteriores, identificar o CI que falhou. Porém, apenas diagnosticar o CI que apresenta a falha não é suficiente, pois esse pode ter falhado por diferentes causas raiz.

Em uma pesquisa recente, Agarwal e Madduri (AGARWAL; MADDURI, 2010) desenvolveram um sistema para identificação automática da causa raiz em falhas assíncronas, geradas nas execuções de mudanças. É importante salientar que essa solução detecta

apenas quebras de SLAs ou falhas em *softwares*, por exemplo, reduções na qualidade de serviços que são monitorados. Além disso, a solução permite detectar falhas assíncronas, ou seja, mesmo que a quebra do SLA não ocorra próxima à execução do CP, essa falha é detectada e analisada. O sistema utiliza associações entre falhas e mudanças para identificar o CP que originou a falha. Essa identificação permite isolar, através da associação, a atividade que gerou a falha. Porém, a atividade pode ter falhado por diferentes motivos, o que resultaria em um grupo de causas raiz. Além disso, a solução limita-se a identificar apenas falhas em *softwares*, não diagnosticando atividades que envolvam outros recursos, por exemplo, recursos humanos.

Apesar de não estarem inseridas na área de ITSM, algumas pesquisas devem ser revisadas, pois fornecem conceitos e propostas que podem ser mapeadas para o Gerenciamento de Problemas. Nesse contexto, Appleby *et al.* (APPLEBY; GOLDSZMIDT; STEINDER, 2001) propuseram o *Yemanja*, um mecanismo baseado na correlação de eventos para diagnosticar falhas multi-camadas. Tal mecanismo objetiva diagnósticos de falhas, em cenários complexos de propagação e pode facilmente correlacionar eventos de redes em baixo nível, como violações na qualidade de serviços. Em um outro trabalho, Steinder e Sethi (STEINDER; SETHI, 2004) propuseram um método para diagnósticos de falhas em serviços fim-a-fim, baseado na análise de sintomas mapeados em grafos de causalidade e suas dependências. A solução permite identificar tanto 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.

Embora a área de ITSM tenha recebido grande atenção da comunidade científica em investigações recentes, nenhuma das pesquisas citadas permite a identificação da causa raiz de falhas, no domínio reativo, ocorridas na execução de mudanças. Além disso, poucos trabalhos exploram a possibilidade de aprendizagem a partir de experiências passadas. Para tratar essas deficiências, nesta dissertação é proposta uma solução conceitual, juntamente com uma expansão do modelo CIM, que permite a identificação da causa raiz de falhas em execuções de CPs. É importante salientar que a solução proposta está situada na área de Gerenciamento de Mudanças, no domínio reativo, em que diferentes fontes de falhas são consideradas que não possibilitam diagnosticá-las de forma automática, por exemplo, erros humanos.

2.5 *Background*

As pesquisas na área de gerenciamento de TI, desenvolvidas pelo grupo de Redes de Computadores da UFRGS, foram materializadas em um sistema prototípico denominado CHANGELEDGE. Na Figura 2.4 é apresentada uma visão geral da arquitetura conceitual desse sistema, destacando os principais componentes, atores envolvidos e as principais interações entre esses elementos. É importante salientar que a visão geral da arquitetura foi elaborada a partir de uma sumarização das soluções propostas em trabalhos anteriores e, portanto, não é contribuição de uma única pesquisa.

O processo é iniciado quando o solicitante da mudança interage com o Sistema de Gerenciamento de Mudanças (*i.e., Change Management System, CMS*) com o intuito de especificar uma RFC. A especificação inicial da RFC é realizada mais precisamente no componente *Change Designer* proposto por Cordeiro *et. al.* (CORDEIRO et al., 2009). Em linhas gerais, uma RFC descreve quais modificações devem ser introduzidas na infraestrutura gerenciada, os CIs afetados (*e.g.*, *firewall*, *switches*, serviços e aplicações) e os propósitos de negócio a serem alcançados. No entanto, não são especificados na RFC

detalhes de como materializar as modificações solicitadas. Posteriormente, um operador interage com o componente *Change Designer* a fim de especificar um plano de mudança preliminar, que conterá detalhes da materialização da RFC especificada anteriormente. Tal plano consiste em um *workflow* de atividades/ações que descrevem, em um alto nível de abstração, como a mudança solicitada pode ser materializada na infraestrutura de TI gerenciada.

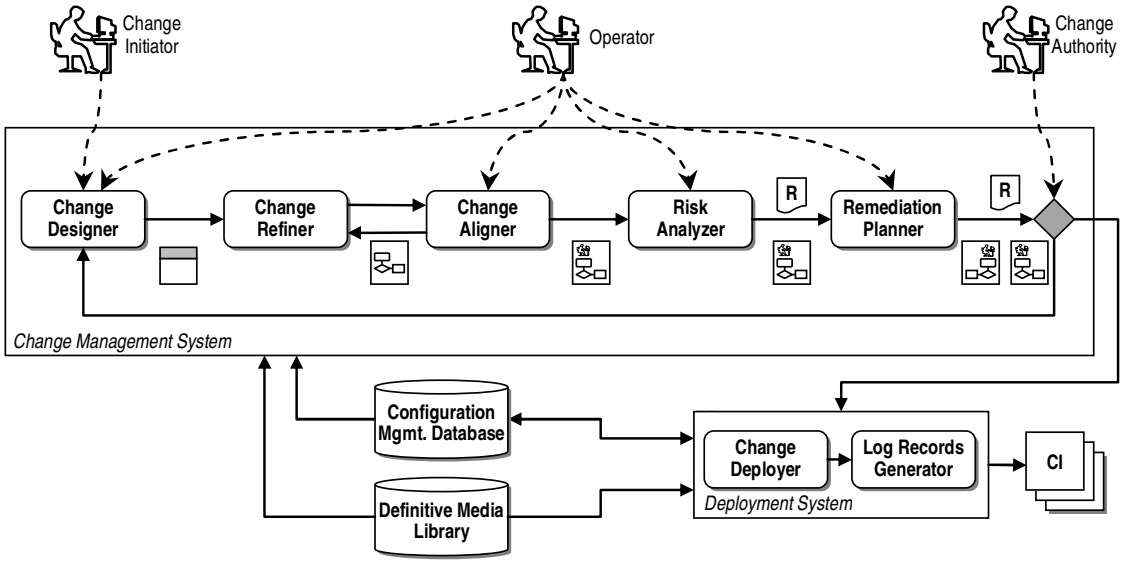


Figura 2.4: Sumarização da arquitetura conceitual do sistema CHANGELEDGE

Em um passo subsequente, o componente *Change Refiner* cria um CP sem a intervenção humana. Tal componente foi inicialmente proposto por Cordeiro *et. al.* (CORDEIRO et al., 2009) e posteriormente aperfeiçoado por Lunardi *et. al.* (LUNARDI et al., 2009). O CP consiste basicamente em um *workflow* de atividades de baixo nível que podem ser executadas diretamente sobre os CIs. Para criar um CP são utilizadas informações sobre: a infraestrutura alvo, oriundas do CMDB; informações sobre *softwares*, disponibilizadas na DML; e detalhes especificados no plano de mudança preliminar. Cada atividade do CP é descrita utilizando a notação *Activity Modeling Notation* (AMN) desenvolvida por Cordeiro *et. al.* (CORDEIRO et al., 2009). Tal notação é utilizada para permitir ao sistema identificar quais os elementos envolvidos e garantir que as atividades especificadas não possuam ambiguidades. Abaixo, algumas representações de atividades na notação AMN.

1. `start Service <serviço> at ComputerSystem <host>`
2. `stop Service <serviço> at ComputerSystem <host>`
3. `install SoftwareElement <software> at ComputerSystem <host> with <parâmetros>`
4. `configure ManagedSystemElement <elemento> of type <tipo> at ComputerSystem <host> using Setting <configurações>`
5. `copy Setting from SoftwareElement <sw> at ComputerSystem <host1> to SoftwareElement <sw> at ComputerSystem <host2>`

Como pode ser observado nos exemplos acima, a notação AMN possibilita representar diferentes instruções. A instrução representada no Exemplo 1 é utilizada para iniciar um serviço, em que `start Service <serviço>` refere-se a um serviço específico que deve ser iniciado, e `at ComputerSystem <host>` indica o sistema computacional no qual esse serviço está instalado. Similarmente, a instrução representada no Exemplo 2 é utilizada para parar um serviço, em que `stop Service <serviço>` indica a ação de parar o serviço especificado, e `at ComputerSystem <host>` refere-se ao sistema computacional no qual esse serviço está instalado.

Algumas instruções possibilitam realizar tarefas mais complexas, por exemplo, instalar ou configurar um *software*. A instrução representada no Exemplo 3 realiza a instalação de um *software*, em que `install SoftwareElement <software>` indica a ação de instalação, bem como qual o *software* a ser instalado. Além disso, a expressão `at ComputerSystem <host>` indica qual o sistema computacional será alvo da instalação solicitada. Adicionalmente, podem ser necessários parâmetros para a instalação desse *software*, em que `with <parâmetros>` indica quais parâmetros devem ser repassados ao programa de instalação. Operações como configuração e cópia de configurações também são possíveis e representadas nos Exemplos 4 e 5, respectivamente. Um conjunto complementar de expressões na notação AMN pode ser encontrado no trabalho de Cordeiro *et. al.* (CORDEIRO et al., 2009).

Após o refinamento da mudança, o CP resultante é alinhado aos objetivos do negócio pelo componente *Change Aligner*, proposto por Lunardi *et. al.* (LUNARDI et al., 2010). Tal componente aloca, quando necessário, recursos humanos às atividades do CP, observando os objetivos do negócio. Para nortear essa alocação, são utilizadas as seguintes métricas: tempo máximo de execução; custo máximo tolerado; esforço humano necessário para execução de uma atividade (*ManPower*); esforço computacional requerido para execução de uma atividade automatizada (*ComputingPower*); e o custo financeiro por hora dos recursos humanos utilizados (*CostperHour*). Através da avaliação dessas métricas e da utilização de uma das diferentes estratégias propostas é possível otimizar, tanto o custo financeiro associado ao uso de recursos humanos, quanto o tempo total de execução de um CP. O resultado desse componente é o mesmo CP, porém, com recursos humanos alocados, bem como alinhado aos objetivos do negócio.

Logo após o alinhamento aos objetivos do negócio, o CP resultante é avaliado quanto ao risco da ocorrência de falhas na sua execução. Tal procedimento foi proposto por Wickboldt *et. al.* (WICKBOLDT et al., 2011) e materializado no componente *Risk Analyzer*. Através da coleta de informações do próprio ambiente de TI, o componente estima os riscos para cada atividade do CP. Tal estimativa é baseada em traços de execuções anteriores desse CP ou de CPs similares, isto é, CPs que contenham atividades idênticas ou similares. A fim de melhorar a precisão do cálculo da similaridade das atividades, Bianchin *et. al.* (BIANCHIN et al., 2010) propuseram um algoritmo que permite calcular a similaridade das atividades de um CP com base em informações de execuções anteriores de CPs similares. Ao final, é gerado um relatório de riscos contendo a quantificação da probabilidade de falhas para cada atividade do CP, bem como o impacto de uma possível falha na execução de uma atividade. Esse relatório permite identificar atividades que necessitam de uma maior atenção na fase de planejamento e servirá de insumo para uma futura aprovação do CP.

Posteriormente, o componente *Remediation Planner*, proposto por Machado *et. al.* (MACHADO et al., 2009), gera automaticamente planos de *rollback* para atividades atômicas baseado em definições do operador. Caso ocorra alguma falha durante a execução

da mudança, tais planos são executados a fim de reverterem as alterações na infraestrutura alvo, fazendo com que a infraestrutura de TI permaneça em um estado consistente. De acordo com a ITIL, planos de remediação são requeridos antes de qualquer mudança ser submetida para execução. É importante salientar que apesar de automatizados, o operador pode interferir ou alterar o resultado de quaisquer desses componentes com o intuito de refletir suas preferências ou decisões.

Por fim, o CP deve ser aprovado por uma autoridade de mudança, normalmente em uma reunião do CAB. Caso não seja aprovado, a CP é retornado ao estado inicial de RFC e deverá passar por todo o processo novamente, visando a atender as melhorias esperadas pela autoridade de mudança. Assumindo que o CP tenha sido aprovado, esse 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 CMDB a fim de manter sempre uma visão atualizada da infraestrutura de TI gerenciada.

É possível observar que o sistema CHANGELEDGE suporta diferentes requisitos relacionados à falhas, por exemplo, estimativa de riscos e planos de *rollback*. Esses requisitos permitem tratar falhas tanto de forma pró-ativa, quanto reativa. Entretanto, apenas analisar os riscos e/ou desfazer as alterações na ocorrência de um erro, não é suficiente para eliminar a causa raiz de falhas. Além disso, a ITIL recomenda fortemente a identificação da causa raiz, a fim de resolver a origem das falhas da melhor maneira possível. Para tanto, nesta dissertação é proposta uma solução conceitual que permite a identificação da causa raiz de falhas na execução de mudanças. Tal solução é apresentada no Capítulo 3.

3 SOLUÇÃO CONCEITUAL

Neste capítulo, a solução conceitual que permite a identificação da Causa Raiz (*i.e., Root Cause*, RC) de problemas ocorridos durante a execução de CPs é apresentada. Em contraste com as propostas existentes, a solução desenvolvida foca na reutilização de informações, coletadas do próprio ambiente de TI, para auxiliarem no diagnóstico de problemas. O reaproveitamento dessas informações permite à solução proposta reutilizar o conhecimento adquirido, bem como se adaptar às nuances da infraestrutura alvo. Ademais, o operador interage com o sistema durante o diagnóstico. Isso permite que as respostas informadas pelo operador sejam consideradas na seleção das próximas perguntas. A fim de facilitar a adoção do sistema, a solução é compatível com o modelo CIM (DMTF, 2008), um padrão amplamente difundido que permite representar infraestruturas de TI.

Nesta dissertação, as falhas ocorridas durante a execução de mudanças são consideradas recorrentes, isto é, ao observar-se o histórico de execuções de um CP e as informações de diagnóstico associadas, é possível analisar falhas passadas e identificar a RC correta. Além disso, a solução desenvolvida foca apenas na identificação das RCs de falhas durante a execução de uma mudança, portanto, as demais etapas do processo de Gerenciamento de Problemas serão tratadas por um sistema ou operador, bem como a determinação da melhor solução para o problema relatado.

Em um primeiro momento, será discutido, na Seção 3.1 o modelo de informação que permite representar as informações necessárias para a implantação da solução proposta. A arquitetura conceitual juntamente com os elementos, atores e interações é apresentada na Seção 3.2. O componente *Root Cause Analyzer* e os algoritmos que possibilitam a identificação da RC são detalhados na Seção 3.3, enquanto que na Seção 3.4 são discutidas diferentes estratégias para a seleção de perguntas, visando a reduzir o número de interações até a identificação da RC correta.

3.1 Modelo de Informação

Conforme apresentado no capítulo anterior, a ITIL recomenda a utilização de um CMDB, a fim de armazenar informações que representem completamente o atual estado da infraestrutura de TI de uma organização. Alguns modelos existentes contemplam boa parte das informações necessárias para descrever precisamente humanos, *softwares*, *hardwares* e demais CIs. O modelo CIM, proposto pelo *Distributed Management Task Force*, é amplamente utilizado pelas organizações para representar infraestruturas de TI (DMTF, 2008).

O modelo CIM permite descrever, de forma detalhada, os elementos de uma infraestrutura de TI, tais como, sistemas computacionais, pessoas e processos, bem como as relações entre esses elementos. Além disso, o CIM possui um conjunto específico de classes que são destinadas à representação de informações sobre suporte, incluindo a documentação de problemas e incidentes. Tais classes são referenciadas como um padrão denominado *Problem Resolution Standard* (PRS). No entanto, não existem classes que permitam representar diagnósticos e RCs. Isso faz com que o conhecimento obtido com diagnósticos anteriores não seja reaproveitado.

Com o objetivo de desenvolver uma solução que auxilie na identificação de RCs de falhas, nesta dissertação é proposta uma extensão do modelo CIM/PRS que permite armazenar informações sobre diagnósticos, bem como associá-las aos CIs que falharam. A extensão do modelo consiste basicamente na inclusão de um subconjunto de classes e seus relacionamentos. Na Figura 3.1 é apresentada uma simplificação do modelo CIM/PRS, bem como a extensão proposta. As classes pertencentes ao modelo CIM/PRS são apresentadas em cinza claro, enquanto que as classes pertencentes à extensão proposta são apresentadas em cinza escuro.

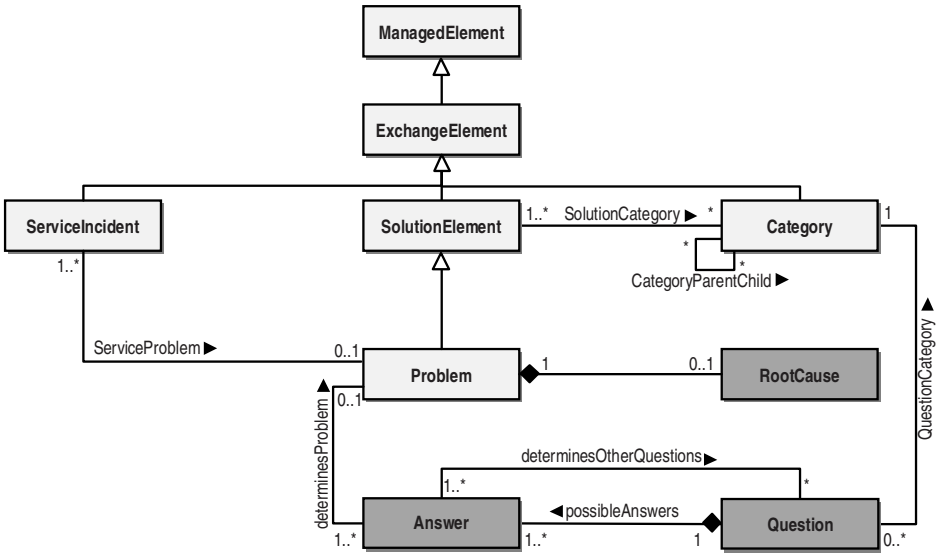


Figura 3.1: Extensão proposta que representa informações sobre diagnósticos interativos

Propostas no modelo CIM/PRS, as classes *Problem* e *ServiceIncident* possibilitam representar, respectivamente, instâncias de problemas e incidentes. É importante salientar a associação existente entre essas classes, pois permite associar diversos incidentes ocorridos a um único problema. Em um caso hipotético, em que a atividade de *verificar o acesso do webmail* falhou, o problema relatado é a *falha ao verificar acesso do webmail*. Por outro lado, cada caso em que o cliente que não consegue acessar seu *webmail* é considerado como um incidente. Isso quer dizer que, caso dois clientes façam reclamações sobre a indisponibilidade do serviço, consequentemente, serão registrados dois incidentes diferentes. Porém, ambos os incidentes estão associados ao mesmo problema. Essa separação facilita o gerenciamento de problemas e de incidentes permitindo aos gerentes direcionarem esforços para os problemas que, segundo os critérios da organização, possuam uma maior prioridade.

As RCs de falhas recorrentes são representadas por instâncias da classe proposta *RootCause*. Além disso, a classe *RootCause* é agregada à classe *Problem* do modelo CIM/PRS. Essa agregação permite ao sistema de diagnóstico vincular uma causa raiz específica tanto

a um problema específico, quanto aos vários incidentes que podem ter ocorrido em decorrência desse problema/falha. No entanto, caso o problema não tenha sua causa raiz identificada, não há nenhuma instância de classe *RootCause* agregada à instância da classe *Problem*.

A solução proposta baseia-se na interatividade entre o operador e o sistema de diagnóstico. Assim sendo, o operador seleciona uma resposta para uma pergunta selecionada pelo sistema. Essa interação repete-se até que uma causa raiz seja identificada. A fim de materializar tal interação, as classes *Question* e *Answer* são necessárias. Uma pergunta que deverá ser respondida pelo operador é representada por uma instância da classe *Question*. Além disso, uma pergunta possui agregadas diversas instâncias da classe *Answer*. Isso se deve ao fato que o operador deve responder à pergunta somente selecionando uma das respostas pré-estabelecidas.

As respostas pré-estabelecidas para uma pergunta são representadas por instâncias da classe *Answer*. Um conjunto de determinadas respostas pode terminar a seleção de mais perguntas ou identificar um problema específico e, consequentemente, a causa raiz agregada ao problema. Além disso, uma resposta está associada a uma única pergunta, ou seja, por mais que duas perguntas diferentes possuam alternativas idênticas, cada uma das alternativas é representada por uma instância diferente da classe *Answer*. Assim, o conjunto de perguntas e respostas que permitem identificar uma RC é único e recebe o nome de conjunto de diagnóstico.

A classe *Category*, proposta no modelo CIM/PRS, permite agrupar instâncias da mesma classe que compartilhem características semelhantes. Perguntas sobre o mesmo assunto são agrupadas através da associação com uma mesma instância da classe *Category*. De acordo com a ITIL, uma categoria pode possuir até três níveis de categorias dependentes, ou seja, quatro níveis de categorias no total. Além das perguntas, os CIs instanciados também estão associados a uma instância da classe *Category*. Isso possibilita identificar, através dos relacionamentos entre esses elementos, quais perguntas são referentes aos CIs identificados, bem como qual o nível de uma pergunta. Dessa forma, as perguntas de aspecto mais geral serão selecionadas primeiro. Por exemplo, o CI *Apache* está associado à categoria *Web Server*. Por sua vez, a categoria *Web Server* é dependente de uma categoria de nível mais alto chamada *Software*. É importante salientar que nesse exemplo, a categoria *Web Server* é considerada de Nível 2, pois é mais específica, e a categoria *Software* pertencente ao Nível 1, pois possui um aspecto mais geral. Além disso, ao selecionar o CI *Apache* as categorias *Software* e *Web Server* também são selecionadas.

O sistema proposto analisa informações sobre diagnósticos anteriores, mapeadas nos *logs* do sistema, para auxiliar no diagnóstico de falhas recorrentes. Através dessa análise é possível estimar quais RCs possuem uma maior probabilidade de serem a real causa da nova falha. Além disso, é possível definir qual a próxima pergunta que deve ser respondida pelo operador. Para tanto, o modelo CIM/PRS propõe um subconjunto de classes de propósito geral que permitem representar *logs* de um sistema. Porém, são necessários relacionamentos entre esse subconjunto de classes e as classes propostas, a fim de armazenarem as informações de diagnósticos anteriores. Na Figura 3.2 são apresentadas as classes que permitem armazenar tais informações. As classes do modelo CIM/PRS são apresentadas em cinza claro, enquanto que as classes propostas são apresentadas em cinza escuro.

Tipicamente, instâncias da classe *LogicalElement* representam componentes lógicos de um sistema que permitem o armazenamento de informações, por exemplo, arquivos ou banco de dados. Além disso, instâncias da classe *EnabledLogicalElement* representam

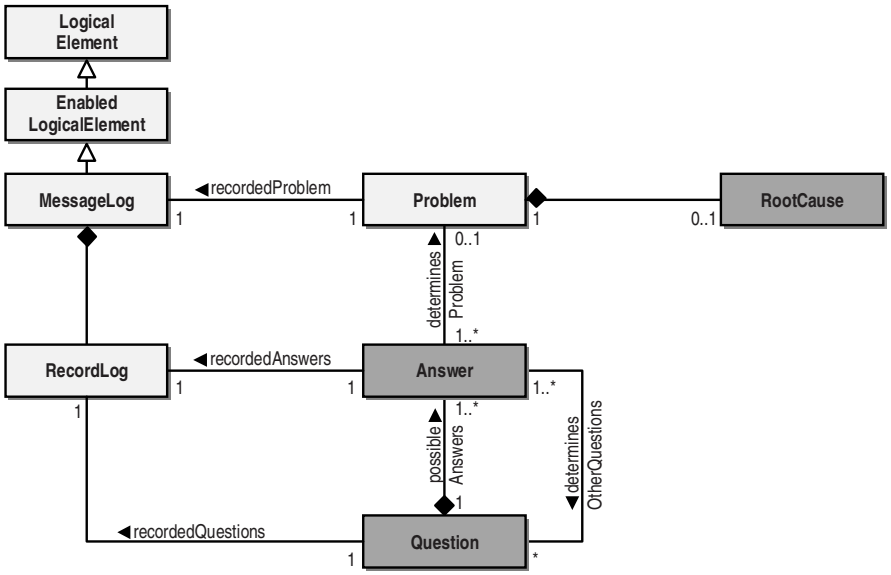


Figura 3.2: Extensão proposta que permite armazenar diagnósticos anteriores nos *logs*

uma abstração de que esses componentes lógicos estão atualmente habilitados e prontos para utilização. Uma instância da classe *RecordLog* contempla uma pergunta selecionada pelo sistema, a categoria dessa pergunta e a resposta informada pelo operador para cada interação. Esse armazenamento a cada interação permite que o operador suspenda o diagnóstico por um determinado período e o retome mais tarde.

Uma instância da classe *MessageLog* representa um diagnóstico completo, pois agrega todas as instâncias de perguntas e respostas para um determinado diagnóstico. Além disso, são representadas nessas instâncias informações sobre o PR, descritas na Seção 2.3, bem como a RC identificada ao final do processo de diagnóstico e a categoria informada. Dessa forma, ao analisar as instâncias da classe *MessageLog*, o sistema proposto possui acesso às informações relevantes sobre os diagnósticos anteriores. Isso possibilita reutilizar o conhecimento adquirido com experiências passadas em novos diagnósticos.

3.2 Arquitetura Conceitual

Com base nos processos do Gerenciamento de Mudanças e do Gerenciamento de Problemas, descritos respectivamente nas Seções 2.2 e 2.3, é proposta uma solução para a identificação da causa raiz de problemas ocorridos na execução de mudanças de TI. Uma visão geral da arquitetura conceitual dessa solução é apresentada na Figura 3.3. O processo de mudança inicia quando um operador interage com um sistema genérico para o Gerenciamento de Mudanças, aqui denominado de Sistema de Gerenciamento de Mudanças (*i.e*, *Change Management System*). Tal interação objetiva, em um primeiro momento, gerar uma RFC, no componente *Change Designer*. Logo após, a RFC é refinada pelo componente *Change Refiner*. O resultado desse refinamento é um *workflow* com atividades de baixo nível que podem ser efetivamente executadas sobre os CIs, denominado CP. Um detalhamento sobre esses componentes, bem como sobre demais componentes que podem ser incorporados ao processo do Gerenciamento de Mudanças, pode ser visto na Seção 2.5.

Posteriormente, o CP é avaliado por uma autoridade de mudança, ocultada a fim de melhorar a legibilidade. Caso reprovado, o CP retorna ao componente *Change Designer*

para possíveis alterações ou ajustes. Se for aprovado, o CP é então encaminhado ao Sistema de Implantação (*i.e.*, *Deployment System*). Em tal sistema, o CP é então executado sobre a infraestrutura de TI a fim de refletir as alterações desejadas. Durante a execução do CP, o Sistema de Implantação armazena informações sobre a execução em *logs*, bem como mantém uma visão atualizada da infraestrutura de TI no CMDB.

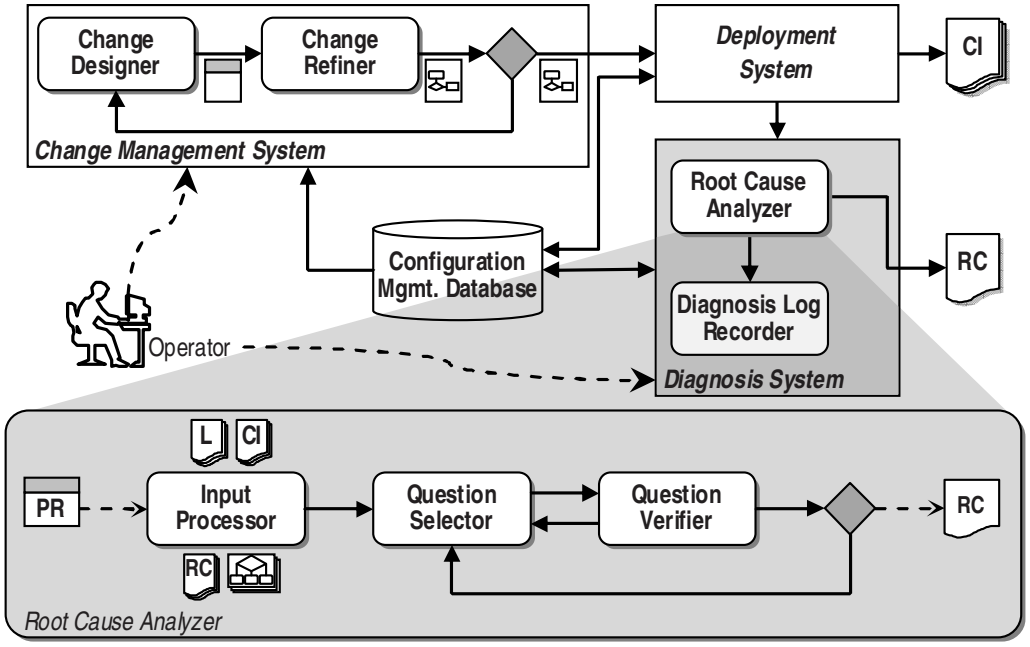


Figura 3.3: Visão geral da arquitetura conceitual da solução proposta

A fim de oferecer suporte à identificação da RC de falhas na execução de CPs, foi introduzido ao fluxo do processo de mudança o Sistema de Diagnóstico (*i.e.*, *Diagnosis System*). Tal sistema somente é acionado caso uma falha seja detectada durante a execução de um CP. É importante salientar que a detecção de falhas pode ser realizada de diversas formas, por exemplo, detecção automática, notificação de um contratante pelo *Service Desk* ou através da análise de relatórios por um operador experiente. A detecção de falhas não é abordada nesta dissertação, mas sim o diagnóstico e a identificação da causa raiz da falha. Para tanto, após a detecção da falha pelo Sistema de Implantação, um PR é gerado, iniciando assim o processo de diagnóstico.

Assim que o PR é gerado, o Sistema de Diagnóstico identifica todas as informações necessárias para determinar a RC da falha reportada. Entre essas informações, oriundas do CMDB, é válido citar: os CIs que podem ter influenciado ou ocasionado a falha; as categorias associadas aos CIs; as RCs associadas às categorias selecionadas; os conjuntos de diagnóstico associados a cada uma das RCs; e por fim, os *logs* de diagnósticos passados, para cada uma das RCs identificadas. Com base nessas informações, o Sistema de Diagnóstico estima, de acordo com a estratégia selecionada, qual pergunta tem uma maior probabilidade de determinar a RC. Tais informações são armazenadas conforme o Modelo de Informação descrito na Seção 3.1.

Ao interagir com o Sistema de Diagnóstico, a fim de responder à pergunta selecionada, um operador seleciona uma alternativa disponível para a pergunta. A seleção de qual pergunta deve ser respondida pelo operador é realizada pelo componente *Root Cause Analyzer* que será mais bem detalhado na Seção 3.3. Baseado na resposta selecionada pelo operador, o *Root Cause Analyzer* faz um refinamento no conjunto das possíveis RCs. As

RCs que não podem ser mais identificadas são desconsideradas, bem como as categorias, os conjuntos de diagnóstico e os *logs* associados a essas RCs. Logo após, o cálculo para determinar a probabilidade das perguntas é refeito, de acordo com a estratégia previamente selecionada. Então, a pergunta com a maior probabilidade de determinar a RC correta é novamente enviada para que o operador a responda. Essa interação repete-se até que uma única RC possa ser determinada, encerrando o diagnóstico interativo.

3.3 *Root Cause Analyzer*

Ao interagir com o Sistema de Diagnóstico, um operador seleciona uma alternativa, dentre as pré-estabelecidas, para a pergunta informada. Esse procedimento interativo se repete até que uma RC seja identificada para a falha relatada. Isso permite ao sistema gerar dinamicamente um *workflow* de diagnóstico a cada execução, com base nas respostas fornecidas pelo operador. Tal *workflow* é composto pelas perguntas realizadas, bem como pelas respostas informadas e representa todos os passos do diagnóstico até a identificação da RC.

O *Root Cause Analyzer* é o componente do Sistema de Diagnóstico responsável pela identificação da RC e, consequentemente, pela seleção das perguntas. Para realizar a seleção de uma pergunta são consideradas as informações contidas em quatro fontes distintas: (*i*) o problema/incidente que está sendo diagnosticado; (*ii*) os CIs, pertencentes a infraestrutura de TI, que podem ter influenciado ou ocasionado a falha; (*iii*) os *logs* de diagnósticos onde as RCs selecionadas foram identificadas; (*iv*) as respostas fornecidas pelo operador; e (*v*) a estratégia utilizada para a seleção das perguntas.

Como pode ser observado na Figura 3.3, para facilitar a compreensão, tal componente é dividido em três módulos: o *Input Processor*, módulo responsável por reunir todas as informações que serão necessárias durante o diagnóstico; o *Question Selector*, módulo responsável por calcular os pesos das perguntas e respostas, bem como selecionar uma pergunta que deverá ser respondida por um operador; e o *Question Verifier*, módulo responsável por verificar se uma pergunta é considerada óbvia. Tais módulos são detalhados a seguir.

3.3.1 *Input Processor*

Com base nas informações contidas no PR, o módulo *Input Processor* identifica todos os elementos da infraestrutura de TI que podem ter causado ou mesmo influenciado na falha. Para tanto, as dependências do CI informado no PR são mapeadas. Cada elemento da infraestrutura que possui uma relação de dependência com o CI, previamente informado no PR, é armazenado em uma lista. Ao final, essa lista conterá todos os CIs que serão utilizados no processo de diagnóstico da falha. É importante salientar que todas as dependências entre os elementos da infraestrutura são mapeadas pelo modelo CIM.

Para uma melhor compreensão, suponha que o CI *Acesso à Página Web* foi reportado em um PR. O *Acesso à Página Web* é um serviço implementado por um *software*, denominado *Apache*. A relação de dependência existente entre esses dois CIs é descrita pelo CIM como *Service Implementation*. Além disso, o *software Apache* está instalado em um *Servidor Web*, caracterizando uma dependência chamada de *Installed Sw Element*. Após mapear as dependências, os CIs *Acesso à Página Web*, *Apache* e *Servidor Web* são armazenados na lista de elementos que podem ter causado ou influenciado na falha. Um conjunto complementar de dependências entre os CIs pode ser encontrado na descrição do modelo CIM (DMTF, 2008).

De posse da lista de CIs, o módulo busca as categorias nas quais cada CI está associada. Assumindo que a categoria identificada possua múltiplos níveis, então todas as categorias superiores serão selecionadas. Analisando o exemplo anterior, o *Apache* pertence à categoria de Nível 1 *Software* e a categoria de Nível 2 *Servidor Web*. Nesse caso, ambas as categorias serão utilizadas no diagnóstico. Da mesma forma, as categorias associadas aos outros CIs presentes na lista também serão selecionadas.

De acordo com o modelo de informação proposto e detalhado na Seção 3.1, os CIs, as RCs e as perguntas estão associados às categorias. Essa associação permite listar todas as possíveis RCs para uma falha ocorrida, de posse apenas das categorias identificadas no passo anterior. Quando um diagnóstico é concluído, a RC identificada é armazenada nos *logs* desse diagnóstico. Através desse atributo é possível listar os diagnósticos anteriores em que uma determinada RC foi identificada. Tal atributo permite que o histórico possa ser consultado e considerado em novos diagnósticos. Dessa forma, após ter listado todas as possíveis RCs, os *logs* de diagnósticos anteriores são identificados.

Cada RC está associada a um único conjunto de diagnóstico, que contém respostas específicas para determinadas perguntas. Para encerrar um diagnóstico corretamente, uma RC necessita ser identificada. Para tanto, o conjunto de diagnóstico, associado à RC, precisa ser satisfeito em sua totalidade ou até que apenas uma RC possa ser identificada. Logo, as perguntas e respostas que identificam cada uma das RCs, identificadas no passo anterior, também são selecionadas. Tanto as informações selecionadas pelo módulo *Input Processor*, quanto às informações contidas no PR são consumidas pelo módulo *Question Selector*, descrito na próxima sub-seção.

3.3.2 *Question Selector*

Para realizar o diagnóstico interativo, proposto nesta dissertação, o sistema necessita selecionar uma pergunta para ser respondida pelo operador. Tal processo baseia-se nas informações obtidas pelo módulo *Input Processor*. Visando a otimizar a identificação da RC para a falha relatada, as perguntas que possuem uma maior probabilidade de identificarem a RC correta devem ser priorizadas. Para tanto, quatro diferentes estratégias foram propostas e são brevemente descritas a seguir. Um maior detalhamento, bem como a descrição de algoritmos e fórmulas são apresentados na Seção 3.4.

- **Seleção de perguntas baseada nos diagnósticos concluídos com sucesso** – São atribuídos pesos às perguntas, respostas e categorias obtidas no módulo anterior. Os pesos calculados por essa estratégia consideram apenas as RCs identificadas com sucesso em cada um dos *logs*, bem como o conjunto de perguntas e respostas associado a essas RC.

- **Seleção de perguntas baseada em todos os diagnósticos do sistema** – Assim como na primeira estratégia são atribuídos pesos aos elementos obtidos no módulo anterior. Porém, os pesos calculados por essa estratégia consideram todos os diagnósticos. Dessa forma, os diagnósticos concluídos incrementam o peso, enquanto que os diagnósticos frustrados decrementam o peso de uma RC.

- **Seleção de perguntas considerando a idade dos diagnósticos** – Da mesma forma que as estratégias anteriores, os elementos recebem os pesos de acordo com a quantidade de diagnósticos. No entanto, além da penalização por diagnósticos frustrados, os diagnósticos são penalizados pela sua idade. Quando mais antigo o diagnóstico, menor o seu valor no cálculo do peso.

- **Seleção de perguntas baseada na popularidade** – Essa estratégia visa a selecionar a pergunta que representa uma maior probabilidade de identificar a RC correta, considerando tanto o peso, quanto a popularidade das perguntas. Dessa forma, a pergunta que estiver presente em uma maior quantidade de conjuntos de diagnóstico, terá prioridade sobre as perguntas que possuírem o mesmo peso.

Independentemente da estratégia selecionada, o módulo *Question Selector* selecionará uma única pergunta que, de acordo com a estratégia, for mais relevante para o diagnóstico. Além disso, esse módulo é responsável por calcular os pesos que serão utilizados pelo módulo *Question Verifier*. De posse dos *logs* de diagnósticos anteriores é possível calcular o peso de uma pergunta, bem como o peso de cada uma de suas respostas. O peso de uma pergunta representa quantas vezes ela foi selecionada pelo sistema. Por outro lado, o peso de uma resposta representa a quantidade de vezes em que um operador selecionou essa alternativa quando consultado.

Adicionalmente, as estratégias utilizam o peso da categoria para uma primeira seleção. O peso de uma categoria representa o somatório do peso das RCs associadas a essa categoria. A análise desse peso permite identificar qual a categoria é mais relevante para o processo de diagnóstico em um determinado momento. É importante salientar que o cálculo dos pesos, que serão consumidos pelo módulo *Question Verifier*, incluem todos os diagnósticos. Portanto, após a seleção de uma pergunta pelo módulo, os pesos da pergunta, bem como de cada uma das respostas pré-estabelecidas são computados de acordo com a segunda estratégia.

3.3.3 *Question Verifier*

Após a seleção de uma pergunta o módulo *Question Verifier* inicia um processo de verificação. Esse processo visa a determinar se a pergunta selecionada é óbvia ou não. Uma pergunta pode ser considerada óbvia através da análise do seu peso, bem como dos pesos de suas respostas. Caso a pergunta seja considerada óbvia, o módulo automaticamente atribui a resposta que possuir o maior peso para a pergunta. Dessa forma, não há a necessidade da intervenção do operador para responder a pergunta. Logo após, o sistema retorna para a seleção de uma nova pergunta pelo módulo *Question Selector*. Porém, caso a pergunta não seja considerada óbvia a intervenção do operador faz-se necessária, e deve ser selecionada uma resposta entre as alternativas disponíveis.

Os passos que permitem determinar se uma pergunta é óbvia são apresentados no Algoritmo 1. Os dados necessários incluem a pergunta selecionada, as respostas associadas e os pesos calculados. A primeira condição que deve ser satisfeita é o peso da pergunta selecionada ser maior ou igual a um determinado *threshold* (Linha 2). Nesta dissertação, o *threshold* considerado é igual a 10. Tal valor foi derivado de experimentos realizados previamente em um ambiente simulado. De acordo com a necessidade de cada organização, esse valor pode ser ajustado a fim de adaptar a solução a diferentes ambientes. O valor atribuído ao *threshold* é de fundamental importância, pois permite que RCs com poucas execuções não sejam consideradas óbvias.

Posteriormente, o peso de cada uma das alternativas disponíveis é verificado. Caso alguma alternativa possua um peso maior ou igual a 80% do peso da pergunta (Linhas 3 e 4), então a pergunta é considerada óbvia. Nesse caso, a alternativa que satisfez essas condições é atribuída como resposta pelo *Question Verifier* (Linha 5). No entanto, se qualquer uma das condições não for satisfeita por nenhuma das alternativas, então é solicitada a intervenção do operador, que deve responder à pergunta optando por um das

alternativas pré-estabelecidas (Linhas 9 e 10). É importante salientar que a porcentagem utilizada neste algoritmo é proposta pela técnica chamada de Análise de Pareto. Tal técnica é apresentada na ITIL e visa a diferenciar causas potenciais das triviais (OGC, 2007b).

Algoritmo 1 Verificação de uma Pergunta

Entradas: Pergunta selecionada, Respostas pré-estabelecidas e seus pesos calculados.
Saídas: R: Resposta para a Pergunta.

```
1:  R ← null
2:  if pesoDaPergunta >= threshold then
3:      for all A ∈ conjunto de respostas de P do
4:          if pesoDeA >= 80% do pesoDaPergunta then
5:              R ← A
6:          end if
7:      end for
8:  end if
9:  if R é null then
10:     R ← OPERATOR_INTERVATION(P)
11: end if
    return R
```

Em um exemplo hipotético, a pergunta: "*A placa de rede está configurada?*" foi utilizada em 23 diagnósticos diferentes. Tal pergunta possui duas alternativas, a primeira alternativa, o *Sim*, foi selecionada pelo operador como resposta em 19 casos. Já a segunda alternativa, o *Não*, foi selecionada em apenas 4 diagnósticos. Nesse exemplo, o peso da pergunta é igual a 23, o peso da alternativa *Sim* é igual a 19 e o peso da alternativa *Não* é igual a 4.

Analisando o exemplo acima, a pergunta: "*A placa de rede está configurada?*" satisfaz a primeira condição e terá suas alternativas analisadas pelo módulo *Question Verifier*, pois o seu peso é maior que o *threshold* definido. Ao analisar as alternativas, observa-se que o peso da alternativa *Sim* representa 82,61% do peso da pergunta. Por outro lado, o peso da alternativa *Não* representa 17,39% do peso total da pergunta. Dessa forma, a alternativa *Sim* representa mais de 80% do peso da pergunta e satisfaz todas as condições necessárias. Então, tal alternativa será atribuída como resposta, evitando assim que o operador tenha que responder a perguntas consideradas óbvias.

Após concluir o diagnóstico, uma determinada RC é identificada. Tal RC é informada ao operador que determinará se o diagnóstico foi concluído com sucesso ou se a RC identificada não é a real causa da falha. Caso a resposta seja afirmativa, o processo de diagnóstico é encerrado. As perguntas, respostas selecionadas, o PR e a RC identificada são armazenados em *log* pelo componente *Diagnosis Log Recorder*. É importante salientar que cada interação entre o operador e o Sistema de Diagnóstico é armazenada nos *logs*. Isso permite realimentar o sistema através da reconstituição dos diagnósticos anteriores, desde a criação do PR até a identificação da RC pelo sistema. Além disso, os *logs* do sistema podem ser úteis para outros módulos ou soluções que utilizem informações sobre esses diagnósticos.

Caso o operador determine que o diagnóstico realizado pelo sistema foi inconclusivo ou que a RC determinada é incorreta, o processo de diagnóstico é reiniciado, desabilitando

o módulo *Question Verifier*. Dessa forma, o sistema evita a retirada de perguntas que considerou óbvias, possibilitando assim a escolha de uma outra alternativa. Ao final do novo diagnóstico, persistindo a resposta do operador, um alerta é gerado para que um especialista revise, tanto as perguntas e respostas utilizadas, quanto as RCs identificadas. Assim, caso exista alguma inconsistência ou melhorias que devam ser implantadas, o especialista executa as alterações necessárias, por exemplo, a inserção de perguntas e/ou respostas que permitam diferenciar RCs conflitantes. Além disso, quando uma nova RC é encontrada, o especialista a insere no CMDB, juntamente com as perguntas e respostas que permitem a identificação. Dessa forma, em um diagnóstico futuro, essas informações estarão disponíveis e poderão ser utilizadas.

3.4 Estratégias aplicadas no módulo *Question Selector*

Como pode ser observado, a solução detalhada ao longo deste capítulo é totalmente modular. Isso significa que independente da estratégia utilizada pelo operador, a saída do módulo sempre será uma pergunta que deve ser enviada ao operador, bem como o peso da pergunta e de cada uma das suas alternativas. Da mesma forma, as estratégias desenvolvidas baseiam-se nas mesmas entradas: os CIs que representam a infraestrutura afetada; os *logs* de diagnósticos passados; as categorias associadas aos CIs; as RCs associadas às categorias; e o conjunto de diagnóstico associado a cada RC.

Para uma melhor compreensão das estratégias desenvolvidas, suponha que os dados extraídos pelo módulo *Input Processor* estão representados na Tabela 3.1. Como pode ser observado, cinco RCs foram selecionadas, bem como as perguntas, respostas e categorias associadas. Cada linha da tabela contém informações sobre a RC correspondente. Dessa forma, a RC_1 possui três perguntas associadas (P_1, P_2 e P_8), três respostas para as respectivas perguntas (R_1, R_3 e R_{15}) e está associada à categoria C_1.

Tabela 3.1: Exemplo de informações sobre RCs utilizadas pelas estratégias para seleção de perguntas

RCs Selecionadas	Perguntas	Respostas	Categorias
RC_1	P_1, P_2, P_8	R_1, R_3, R_{15}	C_1
RC_2	P_4, P_7	R_8, R_{13}	C_1, C_2
RC_3	P_5, P_6	R_{10}, R_{12}	C_1, C_3
RC_4	P_3, P_9	R_5, R_{17}	C_1, C_2
RC_5	P_5, P_{10}	R_{11}, R_{21}	C_1, C_4

Semelhantemente, na Tabela 3.2 é apresentada a quantidade de diagnósticos, em que as RCs previamente selecionadas foram identificadas. Como pode ser observado, são apresentados os somatórios dos diagnósticos concluídos e frustrados divididos em um período de doze meses. Isso permite considerar uma nova métrica, a idade dos diagnósticos. Tal métrica é implementada na estratégia *Seleção de Perguntas Considerando a Idade dos Diagnósticos*. Assim sendo, é possível afirmar que a RC_1 possui no total, 25 diagnósticos concluídos, sendo 1 diagnóstico nos últimos doze meses e 24 anteriores a doze meses. Além disso, a RC_1 possui 12 diagnósticos frustrados, sendo 4 nos últimos doze meses e 8 anteriores a doze meses.

A arquitetura conceitual, bem como os modelos e o algoritmos apresentados neste capítulo formam a base para a solução proposta para a identificação da RC de falhas

Tabela 3.2: Quantidade de Diagnósticos para cada RCs

RCs Selecionadas	Diagnósticos Concluídos		Diagnósticos Frustrados	
	Últimos 12 Meses	Anteriores à 12 Meses	Últimos 12 Meses	Anteriores à 12 Meses
RC_1	1	24	4	8
RC_2	4	15	1	2
RC_3	5	4	2	1
RC_4	12	0	2	0
RC_5	3	2	1	0

em execuções de mudanças. No entanto, diferentes métodos para otimizar a seleção das perguntas que visam a identificar a RC são mapeados em diferentes estratégias. Tais estratégias são implementadas pelo módulo *Question Selector*. Nesta seção, as quatro estratégias desenvolvidas são detalhadas. Além disso, o exemplo descrito acima será utilizado para exemplificar a seleção de perguntas por cada estratégia desenvolvida.

3.4.1 Seleção de perguntas baseada nos diagnósticos concluídos com sucesso

A primeira estratégia desenvolvida representa uma solução simplista para a seleção de perguntas. Em síntese, são atribuídos pesos às perguntas, respostas e categorias baseados no peso das RCs associadas. O peso de uma RC é calculado pelo somatório dos diagnósticos concluídos em que a RC foi corretamente identificada. Caso existam categorias, perguntas ou resposta que estejam associadas a mais de uma RC, o peso final desses elementos é calculado através do somatório do peso de todas as RCs associadas.

O Algoritmo 2 materializa a estratégia proposta para o cálculo dos pesos das categorias, perguntas e respostas com base nos pesos das RCs associadas. Para cada RC selecionada é atribuído o peso baseado nos *logs* (Linha 2). Cada uma das perguntas (Linha 4), respostas (Linha 7), e categorias associadas (Linha 10) tem o peso da RC selecionada acrescido ao seu peso atual. Isso permite que uma pergunta ou categoria que está associada a várias RCs tenha um peso mais elevado do que uma que esteja associada a apenas uma dessas RCs. No entanto, o peso da RC será adicionado ao peso da resposta correspondente apenas se essa estiver presente no conjunto de diagnóstico da RC analisada.

Analisando o exemplo descrito anteriormente, o peso da RC_1 é igual a 25. Esse peso deriva da quantidade de diagnósticos concluídos em que tal RC foi identificada. Dessa forma, o peso de cada uma das perguntas e categorias associadas a essa RC é acrescido em 25 unidades. É importante salientar que se uma pergunta, categoria ou resposta estiver associada a mais de uma RC, os pesos são somados e atribuídos aos elementos correspondentes. Pode ser observado que a categoria C_1 está associada a todas as RCs, então o peso dessa categoria será o somatório dos pesos de todas as RCs. Nesse caso, o peso da categoria C_1 é igual a 70. De forma semelhante, a pergunta P_5 possui como peso o valor 14. Tal valor é o resultado do somatório dos pesos da RC_3 e RC_5, as quais estão associadas à pergunta P_5 .

Após a atribuição dos pesos para todas as perguntas, respostas e categorias associadas às RCs selecionadas anteriormente, o módulo *Question Selector* seleciona uma pergunta para ser respondida pelo operador. Essa seleção é realizada com base nos pesos. Primeiramente, é selecionada a categoria que possui o maior peso. Dentro dessa categoria, a pergunta com o maior peso é selecionada. Quando existir mais de uma pergunta com o

Algoritmo 2 Atribuição dos Pesos

Entradas: S: Conjunto de RCs possíveis, conjunto de diagnóstico e categorias associadas para as CIs identificadas; $Logs$: *logs* de diagnósticos anteriores.

Saídas: $peso$: Array que contém o peso calculado para os diferentes elementos.

```
1: for all RC ∈ conjunto de RCs de S do
2:     pesoRC ←WEIGHT_CALCULATE(RC,Logs)
3:     for all Pergunta ∈ conjunto de diagnóstico da RC do
4:         peso[Perguntas][Pergunta] ←WEIGHT_ADD(Pergunta,pesoRC)
5:     end for
6:     for all Resposta ∈ conjunto de diagnóstico da RC do
7:         peso[Respostas][Resposta] ←WEIGHT_ADD(Resposta,pesoRC)
8:     end for
9:     for all Categoria ∈ conjunto de categorias da RC do
10:        peso[Categorias][Categoria] ←WEIGHT_ADD(Categoria,pesoRC)
11:    end for
12: end for
    return peso
```

mesmo peso, a pergunta que for de maior nível é escolhida, ou seja, a pergunta que não depender de nenhuma outra ou que for associada à categoria de nível mais alto.

Após computar os pesos de todos elementos utilizando essa estratégia, observando os valores detalhados nas Tabelas 3.1 e 3.2, o módulo *Question Selector* atribui os pesos às perguntas. Assim sendo, as perguntas são classificadas, pelo peso, na ordem conforme segue: em primeira ordem, as perguntas P_1, P_2 e P_8 que possuem peso 25; em segunda ordem, as perguntas P_4 e P_7 com peso igual a 19; em terceira ordem, a pergunta P_5 possui peso igual a 14; em quarta ordem, as perguntas P_3 e P_9 que possuem o peso igual a 12; em quinta ordem, a pergunta P_6 possui peso igual a 9; e, por fim, a pergunta P_{10} tem peso igual a 5. Por outro lado, as categorias também possuem os pesos atribuídos conforme segue: a categoria C_1 possui o peso igual a 70; a categoria C_2 possui peso igual a 31; a categoria C_3 com peso igual a 9; e, por fim, a categoria C_4 com peso igual a 5. Dessa forma, as perguntas associadas à categoria C_1 devem ser selecionadas, pois essa categoria possui o maior peso. Associadas a essa categoria, as perguntas de primeira ordem (P_1, P_2 e P_8) possuem preferência para seleção. Porém, as perguntas P_2 e P_8 são dependentes da pergunta P_1. Portanto, a pergunta selecionada pelo módulo para ser enviada ao operador é P_1.

3.4.2 Seleção de perguntas baseada em todos os diagnósticos do sistema

A estratégia apresentada anteriormente, apesar de simplista, representa uma abordagem coerente e dinâmica. No entanto, tal estratégia tende a viciar o diagnóstico, pois o sistema prioriza determinadas RCs. Isso ocorre quando uma ou mais RCs possuem um número maior de diagnósticos corretos em comparação com as demais selecionadas. Caso a RC correta possua um menor peso, o diagnóstico produzirá uma maior quantidade de interações, entre o operador e o sistema. Com o objetivo de reduzir o número de interações, bem como reutilizar o conhecimento adquirido em todos os diagnósticos, sendo concluídos ou frustrados, uma segunda estratégia foi proposta. Tal estratégia baseia o cálculo dos pesos em todos os diagnósticos registrados nos *logs* do sistema, independente se os diagnósticos foram corretos ou frustrados.

Semelhantemente à primeira estratégia proposta, os pesos são calculados e atribuídos para as perguntas, respostas e categorias obtidas pelo módulo *Input Processor*. Os pesos atribuídos são calculados com base no peso de cada RC associada. Porém, essa estratégia considera todos os diagnósticos, ou seja, RCs que possuem diagnósticos frustrados são penalizadas. Um diagnóstico é considerado frustrado, quando o sistema utiliza pelo menos uma pergunta associada à determinada RC e, ao final do processo interativo, outra RC é identificada como a real causa da falha relatada. Dessa forma, ao término do processo de diagnóstico, o histórico de diversas RCs receberá um diagnóstico frustrado, porém, o diagnóstico concluído será atribuído apenas ao histórico da RC identificada. Assim sendo, o peso de uma RC é calculado pelo somatório dos diagnósticos concluídos, contudo, subtraindo o somatório dos diagnósticos frustrados. Por consequência, os pesos dos demais elementos são reduzidos, pois computam o peso das RCs associadas.

Os passos detalhados no Algoritmo 2 são novamente utilizados para a atribuição dos pesos as categorias, perguntas e respostas. Isso ocorre pois a principal diferença dessa estratégia é o cálculo do peso das RCs e não a atribuição desses pesos aos demais elementos. Além disso, a abordagem para seleção da pergunta a ser respondida pelo operador permanece inalterada. Primeiramente, é selecionada a categoria que possui o maior peso. Associada a essa categoria, a pergunta com o maior peso é selecionada. Caso exista mais de uma pergunta com o mesmo peso, a pergunta de maior nível é escolhida. Entretanto, caso constem diagnósticos frustrados nos *logs* do sistema, os pesos tendem a ser diferentes, resultando assim na seleção de perguntas diferentes em relação à primeira estratégia apresentada.

Para um melhor entendimento, observe novamente as Tabelas 3.1 e 3.2. Ao utilizar a estratégia para seleção de perguntas baseada em todos os diagnósticos, os pesos são atribuídos às categorias e essas ordenadas conforme segue: a categoria C_1 com o peso igual a 49; a categoria C_2 tem o peso igual a 26; a categoria C_3 o peso igual a 6; e, por fim, a categoria C_4 possui o peso igual a 4. Porém, como pode ser observado, a ordem das categorias permanece a mesma. É importante salientar que essa ordem pode ser alterada, resultando na seleção de perguntas associadas a outras categorias. Da mesma forma, as perguntas são classificadas, pelo peso, na seguinte ordem: em primeira ordem, as perguntas P_4 e P_7 que possuem peso 16; em segunda ordem, as perguntas P_1, P_2 e P_8 com peso igual a 13; em terceira ordem, as perguntas P_5, P_3 e P_9 que possuem peso igual a 10; em quarta ordem, a pergunta P_6 com peso igual a 6; e por fim, a pergunta P_{10} tem peso igual a 4.

Considerando os pesos atribuídos anteriormente para as categorias e perguntas, as perguntas associadas à categoria C_1 devem ser selecionadas. Isso se deve ao fato dessa categoria possuir o maior peso dentre as categorias previamente selecionadas. Dentre as perguntas de primeira ordem, apenas a pergunta P_4 está associada à categoria selecionada C_1, visto que a pergunta P_7 está associada à categoria C_2. Nesse caso, não há nenhuma outra pergunta com o mesmo peso e associada à mesma categoria, então a pergunta enviada ao operador pelo módulo é a P_4. Em uma comparação simples com a estratégia anterior, pode ser observado que tanto os pesos das perguntas e categorias, quanto a pergunta selecionada pelo módulo são diferentes. A estratégia desenvolvida permite à solução adaptar-se melhor às nuances dos diagnósticos anteriores, visto que considera tanto os diagnósticos concluídos, quanto os diagnósticos frustrados.

3.4.3 Seleção de perguntas considerando a idade dos diagnósticos

Embora as estratégias introduzidas e apresentadas até então identifiquem, de maneira interativa, a RC de falhas na execução de mudanças, a idade dos diagnósticos não é considerada na seleção das perguntas. Supõe-se que o sistema proposto esteja implantado em uma organização há seis meses. Decorrido esse período, diversos diagnósticos foram realizados e, por consequência, o peso de determinadas RCs está inflacionado em comparação com seu peso inicial. Dessa forma, as RCs recentemente armazenadas no CMDB dificilmente terão suas perguntas selecionadas para serem respondidas pelo operador. Assim sendo, a utilização da métrica idade na seleção de perguntas é de vital importância, pois evita a inflação dos pesos. Por conseguinte, as perguntas associadas às RCs recentemente armazenadas no CMDB possuem uma maior probabilidade de serem selecionadas.

Nessa terceira estratégia proposta, além da penalização por diagnósticos frustrados, os diagnósticos são penalizados pela sua idade. Quando mais antigo o diagnóstico, menor o seu valor no cálculo do peso. Tal penalização é aplicada ao peso tanto de diagnósticos frustrados, quanto de diagnósticos concluídos. A operação que permite calcular o peso de um elemento, com base nas heurísticas aqui definidas é formalizada na Equação 3.1, em que: (*i*) a idade de um diagnóstico pode variar entre 1 e 10 sendo representada por i; (*ii*) a subtração entre 100% e o percentual de penalização que será aplicado aos diagnósticos de uma determinada idade é representada por β_i; (*iii*) a quantidade de diagnósticos concluídos que possuem uma determinada idade são representados por α_i; e, por fim, (*iv*) a quantidade de diagnósticos frustrados de uma idade i é representada por ω_i.

$$\textit{pesoElemento}_{(x)} = \sum_{i=1}^{10} \beta_i \times (\alpha_i - \omega_i) \tag{3.1}$$

É importante salientar que os pesos das RCs são reduzidos através da aplicação de um percentual de penalização conforme a idade dos diagnósticos. Além disso, não é aplicado apenas um percentual de penalização ao peso da RC. Caso existam diagnósticos com diferentes idades, cada idade receberá o percentual de penalização adequado, conforme apresentado na Tabela 3.3.

Tabela 3.3: Percentual de penalização para o cálculo do peso das RCs com base na idade dos diagnósticos

Idade	Tempo do Diagnóstico	Percentual de Penalização
1ª	Até 120 dias	Não se aplica
2ª	De 121 dias até 150 dias	10%
3ª	De 151 dias até 180 dias	20%
4ª	De 181 dias até 210 dias	30%
5ª	De 211 dias até 240 dias	40%
6ª	De 241 dias até 270 dias	50%
7ª	De 271 dias até 300 dias	60%
8ª	De 301 dias até 330 dias	70%
9ª	De 331 dias até 360 dias	80%
10ª	A partir de 360 dias	90%

Desse modo, supõe-se que a RC *placa de rede com defeito* possua a seguinte caracterização de seus *logs*: (*i*) ao todo, 25 diagnósticos concluídos; (*ii*) no total, 15 diagnósticos frustrados; (*iii*) os diagnósticos concluídos estão assim distribuídos, 10 classificados na 1ª idade, 5 classificados na 4ª idade, outros 5 classificados na 7ª idade e, por fim, 5 classificados na 10ª idade; e, (*iv*) os diagnósticos frustrados estão assim distribuídos, 4 classificados na 1ª idade, 6 classificados na 3ª idade, 2 classificados na 8ª idade e, por fim, 3 classificados na 10ª idade. Ao empregar a Equação 3.1, obtém-se o seguinte cálculo $\text{pesoElemento}_{(RC)} = 100\%(10-4) + 90\%(0-0) + 80\%(0-6) + 70\%(5-0) + 60\%(0-0) + 50\%(0-0) + 40\%(5-0) + 30\%(0-2) + 20\%(0-0) + 10\%(5-3)$. Após realizar as devidas operações é computado o peso final no valor de 6,3 para a RC *placa de rede com defeito*.

Utilizando as informações, contidas nas Tabelas 3.1 e 3.2, é possível calcular os pesos das RCs penalizando os diagnósticos antigos. Baseado-se nos pesos das RCs são atribuídos os pesos das categorias conforme segue: a categoria C_1 com peso igual a 18,4; a categoria C_2 possui peso igual a 14,3; a categoria C_3 com peso 3,3; e, por fim, a categoria C_4 com peso 2,2. Ademais, as perguntas são ordenadas, pelo peso, conforme segue: em primeira ordem, as perguntas P_3 e P_9 que possuem peso 10; em segunda ordem, a pergunta P_5 com peso igual a 5,5; em terceira ordem, as perguntas P_4 e P_7 que possuem peso igual a 4,3; em quarta ordem, a pergunta P_6 com peso igual a 3,3; em quinta ordem, a pergunta P_{10} que possui peso igual a 2,2; e por fim, as perguntas P_1, P_2 e P_8 com peso igual a -1,4. Com base nos pesos calculados, o sistema seleciona a categoria C_1. Dentre as perguntas de primeira ordem, P_3 e P_9 apenas a pergunta P_3 está associada à categoria C_1. Portanto, a pergunta P_3 é selecionada para ser enviada ao operador.

Ao analisar os pesos distribuídos por essa estratégia, é válido ressaltar duas importantes observações. A primeira é a drástica redução dos pesos das categorias em relação às estratégias apresentadas até aqui. Tal redução evita que os pesos dos elementos sejam inflacionados à medida que o sistema é utilizado. Isso permite que o sistema convirja rapidamente para a seleção de perguntas associadas às novas RCs, incrementando tanto a dinamicidade, quanto a capacidade do sistema em adaptar-se as mudanças ocorridas no ambiente de TI. A segunda observação, refere-se ao peso negativo apresentado pelas perguntas P_1, P_2 e P_8. Tal peso é possível devido à quantidade de diagnósticos frustrados e não recebe nenhum tratamento especial pelo sistema.

3.4.4 Seleção baseada na popularidade das perguntas

As estratégias apresentadas até aqui, selecionam a pergunta que possui o maior peso calculado. Tal peso é diretamente proporcional à quantidade de diagnósticos concluídos. Porém, quando é selecionada uma RC com um peso muito superior às demais, as estratégias desenvolvidas tendem a escolher apenas perguntas do conjunto de diagnóstico dessa RC. Assim, perguntas pertencentes ao conjunto de diagnóstico de RCs novas ou com pesos inferiores dificilmente serão escolhidas. Visando a amenizar esse problema, a quarta estratégia proposta adiciona uma nova métrica no cálculo dos pesos, a popularidade das perguntas. Assim, o peso das perguntas associadas a um grande número de RCs será incrementado e, por consequência, a probabilidade dessas perguntas serem escolhidas será maior.

Nessa estratégia, o cálculo do peso das perguntas é proposto de forma a ponderar igualmente tanto o peso, quanto a popularidade da pergunta, conforme detalhado na Equação 3.2. Tal equação considera diferentes fatores, onde: (*i*) a popularidade de uma pergunta é obtida através da razão entre α_x, que representa a quantidade de ocorrências da

pergunta x nos conjuntos de diagnósticos de todas as RCs selecionadas, e n que representa a número total de RCs selecionadas; (*ii*) a probabilidade de identificação de uma RC é representada por β_{RCi}, tal valor é obtido através da razão entre o peso da RC analisada e o peso total das RCs associadas às categorias selecionadas; e (*iii*) a quantidade de ocorrências da pergunta x no conjunto de diagnóstico associado a uma determinada RC é representada por $\alpha_{RCi,x}$.

$$pesoPergunta_{(x)} = \frac{\frac{\alpha_x}{n} + \sum_{i=1}^{n} \beta_{RCi} \times \alpha_{RCi,x}}{2} \tag{3.2}$$

Assim como na segunda estratégia proposta, os pesos calculados consideram tanto os diagnósticos concluídos, quanto os diagnósticos frustrados. Porém, como pode ser observado na Equação 3.2, o peso das perguntas é obtido através de uma operação de soma entre a popularidade de uma pergunta e o somatório das probabilidades de cada RC multiplicado pela quantidade de ocorrências da pergunta no conjunto de diagnóstico dessa RC. Ao final, o valor computado para essa operação é dividido por 2. Como resultado obtém-se um valor entre 0 e 1 que representa o peso calculado para a pergunta. É importante salientar que cada pergunta pode estar presente no máximo uma única vez em cada conjunto de diagnóstico. Assim sendo, caso $\alpha_{RCi,x}$ seja igual a 0, a probabilidade da RC avaliada não é utilizada no somatório. Por outro lado, caso $\alpha_{RCi,x}$ seja igual a 1, a probabilidade dessa RC é utilizada no somatório e, por consequência, utilizada no cálculo do peso da pergunta.

Utilizando as informações, detalhadas nas Tabelas 3.1 e 3.2, é possível empregar a Equação 3.2 e computar os pesos para as perguntas, segundo a estratégia aqui desenvolvida. Porém, os pesos das RCs e categorias são calculados conforme a segunda estratégia e são ordenados como segue: a categoria C_1 com o peso igual a 49; a categoria C_2 tem o peso igual a 26; a categoria C_3 o peso igual a 6; e, por fim, a categoria C_4 possui o peso igual a 4. Ademais, aplicando a equação, detalhada anteriormente, obtém-se os pesos de cada pergunta considerando a popularidade dessas. As perguntas são ordenadas pelo peso como segue: em primeira ordem, a pergunta P_5 que possuem peso 0,30204; em segunda ordem, as perguntas P_4 e P_7 com peso igual a 0,26327; em terceira ordem, as perguntas P_1, P_2 e P_8 que possuem peso igual a 0,23265; em quarta ordem, as perguntas P_3 e P_9 com peso igual a 0,20204; em quinta ordem, a pergunta P_6 que possui peso igual a 0,16122; e por fim, a pergunta P_{10} com peso igual a 0,14082.

Observando os pesos calculados, a categoria selecionada pelo sistema para iniciar o diagnóstico será a C_1. Dentre as perguntas, apenas uma é categorizada como de primeira ordem, a pergunta P_5 e está associada à categoria C_1. Portanto, a pergunta P_5 é selecionada para ser enviada ao operador. É válido lembrar que a equação detalhada nesta subseção visa a calcular apenas o peso das perguntas. Assim sendo, o cálculo dos pesos das RCs, bem como das categorias é computado de acordo com a segunda estratégia apresentada. Além disso, o peso computado para as perguntas é utilizado somente para a seleção de perguntas. Portanto, a verificação de perguntas óbvias, realizada pelo módulo *Question Verifier*, utiliza os pesos calculados considerando todos os diagnósticos, conforme descrito na Subseção 3.4.2.

Em síntese, ao longo deste capítulo foram apresentados a extensão do modelo de informação, a solução conceitual e os algoritmos que permitem realizar um diagnóstico interativo visando a identificar a RC de uma falha. Além disso, são detalhadas diferentes estratégias que permitem selecionar uma pergunta para ser respondida pelo operador. Tal seleção é realizada por cada estratégia desenvolvida utilizando diferentes métricas. Assim, com base nas mesmas informações é possível selecionar perguntas diferentes. Como pode ser visto no decorrer da Seção 3.4, cada estratégia desenvolvida escolheu uma pergunta diferente para o mesmo exemplo. Isso evidencia que cada estratégia possui pontos fortes e fracos, sendo que cada uma é indicada para diferentes situações. A seguir, no Capítulo 4, são conduzidos dois diferentes estudos de caso que permitem evidenciar as principais características da solução desenvolvida, detalhada neste capítulo.

4 AVALIAÇÃO

Apesar de extremamente comuns no ambiente corporativo, mudanças realizadas sobre a infraestrutura de TI são suscetíveis a diversas falhas. Tais falhas podem resultar na redução da qualidade, ou em casos mais extremos, na interrupção dos serviços prestados pelas organizações. Nesse contexto, o sistema CHANGELEDGE, detalhado na Seção 2.5 e desenvolvido pelo Grupo de Redes da UFRGS, foi proposto para automatizar o planejamento e a implantação de mudanças. O Sistema de Diagnóstico, apresentado no Capítulo 3, foi materializado a fim de oferecer suporte à identificação de causas raiz de falhas na execução de mudanças realizadas pelo CHANGELEDGE.

Para provar o conceito e a viabilidade técnica da solução proposta, neste capítulo, dois estudos de caso são apresentados. O primeiro estudo de caso, detalhado na Seção 4.1, objetiva evidenciar as principais características da solução proposta, tais como: (*i*) ser adaptável à infraestrutura onde a falha ocorre; (*ii*) detectar diferentes RCs a partir de uma mesma falha; e, (*iii*) considerar as respostas do operador durante o diagnóstico. O segundo estudo de caso, apresentado na Seção 4.2, objetiva salientar as diferenças entre as quatro estratégias propostas na Seção 3.4, bem como demonstrar o correto funcionamento da solução considerando um ambiente corporativo real. Como resultado geral de ambos os estudos de caso, foi observada a identificação da RC de falhas ocorridas durante a execução de mudanças, assim como as demais características da solução, entre elas, a flexibilidade, a interação, o reúso do conhecimento e a compatibilidade com os padrões atualmente utilizados.

4.1 Primeiro Estudo de Caso

Como mencionado na Seção 2.5, uma mudança na infraestrutura de TI é iniciada através da especificação de uma RFC. Neste primeiro estudo de caso, a RFC especificada tem o propósito de instalar um serviço de *webmail*, provido pelo *software* Squirrelmail. Logo após, tal RFC é refinada pelo sistema CHANGELEDGE resultando em um CP. As principais atividades envolvidas nesse CP são apresentadas na Figura 4.1. Como pode ser observado, a primeira atividade a ser executada é a instalação do Debian GNU/Linux, um sistema operacional que fornecerá as bases para os demais *softwares*. Posteriormente, são realizadas a instalação e configuração de um serviço de páginas *web*, provido pelo Apache, e um serviço de troca de *e-mails*, provido pelo Exim. É importante observar que essas atividades são realizadas em paralelo. Além disso, verificações dos serviços fornecidos por esses *softwares* são realizadas após a atividade de instalação/configuração. Por fim, o Squirrelmail é instalado e configurado, bem como o serviço provido é verificado em uma atividade posterior.

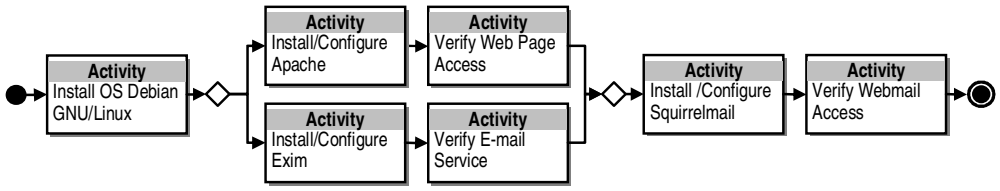
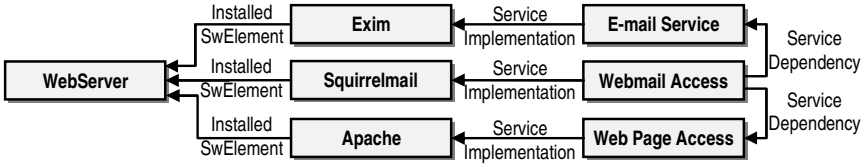
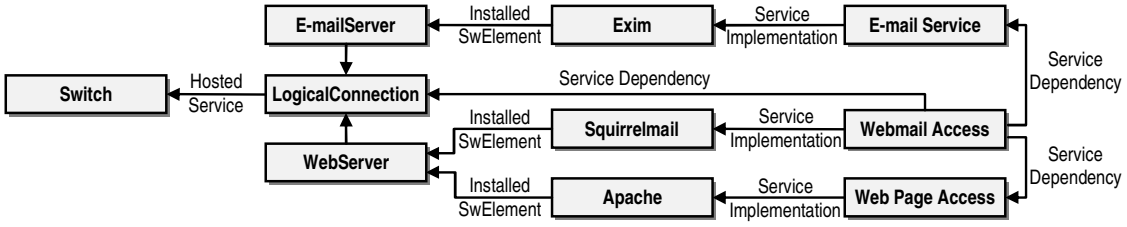


Figura 4.1: *Change Plan* para a instalação de um serviço de *webmail*

Uma das principais características da solução proposta é a capacidade de adaptar-se às nuances da infraestrutura de TI. A fim de evidenciar tal característica, o primeiro estudo de caso é divido em dois cenários diferentes. No Cenário 1, as atividades do CP são executadas em apenas um servidor, denominado *WebServer*. Por outro lado, no Cenário 2, as atividades relacionadas ao *software* Exim são executadas em um servidor denominado *E-mailServer*. Enquanto que em um outro servidor, denominado *WebServer*, as atividades referentes aos *softwares* Apache e Squirrelmail são executadas. As infraestruturas para os Cenários 1 e 2 são mapeadas, respectivamente, nas Figuras 4.2(a) e 4.2(b). Assim sendo, as infraestruturas disponíveis para a execução do mesmo CP são diferentes, possibilitando a ocorrências de falhas distintas. Porém, em ambos os cenários a ocorrência da falha foi relatada na atividade *Verify Webmail Access*.



(a) Infraestrutura com um único servidor



(b) Infraestrutura com dois servidores

Figura 4.2: Instanciação da infraestrutura de TI

De acordo com o processo detalhado na Seção 3.2, quando a falha na atividade *Verify Webmail Access* é detectada, um PR é criado a fim de diagnosticar a causa da falha reportada. Com base nas informações contidas no PR, detalhadas na Seção 2.3, o módulo *Input Processor* identificará o conjunto de CIs dependentes relacionados com o CI *Webmail Access*, no qual a falha foi detectada. Nas Figuras 4.2(a) e 4.2(b) podem ser observadas algumas dependências que permitem identificar os CIs relacionados com a falha. Dentre as dependências aplicadas neste primeiro estudo de caso, estão: (*i*) *Service Dependency*, representa a dependência incondicional para o funcionamento correto de serviços; (*ii*) *Service Implementation*, representa a dependência que um serviço tem perante um *software*; (*iii*) *InstalledSwElement*, representa a dependência da instalação de

um *software* em um computador; e, (*iv*) *Hosted Service*, representa a dependência de um serviço por um *hardware*. É importante observar que essas dependências estão mapeadas no modelo CIM. Porém, existem inúmeras classes que permitem representar as mais diferentes dependências. Um conjunto complementar de dependências entre os CIs pode ser encontrado na descrição do modelo CIM (DMTF, 2008).

Após os CIs dependentes serem localizados, o módulo *Input Processor* seleciona as categorias associadas a cada CI. Com base nessas categorias, as possíveis RCs são identificadas. De posse dessas informações, o módulo seleciona o conjunto de diagnóstico associado a cada RC, bem como os *logs* de diagnósticos anteriores, onde tais RCs foram identificadas. Como a comparação entre as estratégias desenvolvidas não está incluída nos objetivos deste primeiro estudo de caso, os pesos dos elementos são calculados utilizando a estratégia para *seleção de perguntas baseada em todos os diagnósticos do sistema*, detalhada na Subseção 3.4.2. Assim sendo, o módulo *Question Selector* computa o valor dos pesos das RCs considerando tanto os diagnósticos corretos, quanto os diagnósticos frustrados. Baseado nos pesos das RCs, os demais elementos associados são também valorados. Na Tabela 4.1 são representados os CIs e as categorias selecionadas com os seus respectivos pesos calculados para o Cenário 1. Analogamente, na Tabela 4.2, as mesmas informações são apresentadas para o Cenário 2.

Tabela 4.1: Elementos e 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

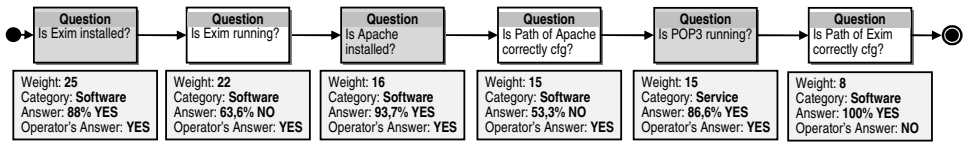
Ao observar a Tabela 4.1, nota-se que o CI *E-mail Service* pertence à categoria de Nível 2 denominada *E-mail*. Além disso, essa categoria está associada a uma categoria de Nível 1 denominada *Service*. Ademais, é possível observar que a categoria *E-mail* possui peso igual a 17, enquanto que a categoria *Service* possui peso igual a 25. É válido mencionar que quando uma categoria é analisada, o peso atribuído é o resultado da soma dos pesos das RCs associadas a essa categoria. Assim sendo, o somatório dos pesos das RCs associadas à categoria de Nível 2 *E-mail* é igual a 17. Porém, quando uma categoria Nível 2 está associada a uma categoria Nível 1, o peso atribuído a categoria de Nível 1 é o somatório dos pesos das RCs associadas à ambas as categorias.

Após atribuir o peso a todos os elementos previamente selecionados, o módulo *Question Selector* seleciona uma pergunta que deve ser respondida pelo operador. Para tanto, a categoria Nível 1 com o maior peso é selecionada. 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 salientar que os pesos são recalculados a cada interação e, portanto, tal ordem pode ser alterada na próxima interação. Comparativamente, o Cenário 2 possui um número maior de CIs dependentes que o Cenário 1 e, consequentemente, uma lista de categorias, perguntas, respostas e RCs mais extensa.

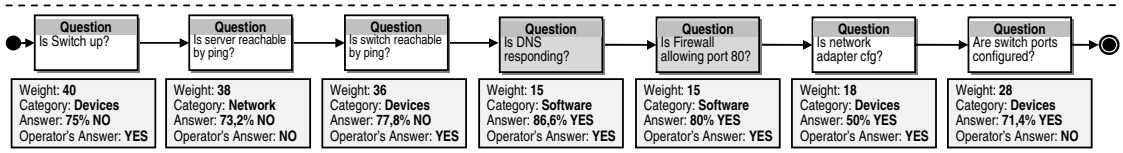
Tabela 4.2: Elementos e seus pesos associados para o Cenário 2

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

Logo após, o módulo *Question Selector* seleciona uma pergunta, com base nos critérios definidos na estratégia selecionada. Para o Cenário 1, tal pergunta é associada à categoria de Nível 1 *Software*, enquanto que no Cenário 2, a pergunta selecionada está associada à categoria de Nível 1 *Devices*. Posteriormente, o módulo *Question Verifier* verifica a obviedade da pergunta selecionada. Caso não seja considerada óbvia, a pergunta é enviada ao operador. Baseado na resposta do operador, o conjunto de possíveis RCs é refinado. Nesse refinamento, as RCs que não possam mais ser identificadas são descartadas, bem como os conjuntos de diagnóstico associados. Caso uma categoria não possua nenhuma RC que possa ser seleciona, tal categoria também é descartada do processo de diagnóstico. Esse processo é repetido até que seja identificada uma RC. Então, a RC identificada é informada ao operador que determina o sucesso do diagnóstico.



(a) Cenário 1



(b) Cenário 2

Figura 4.3: Workflows de diagnóstico gerados pela solução

Os resultados obtidos, durante as interações do processo de diagnóstico, são representados através de um *workflow* de diagnóstico. Tal *workflow* apresenta diversas informações, entre elas: (*i*) as perguntas selecionadas; (*ii*) as respostas informadas, representadas na linha *Operator's Answer*; (*iii*) os pesos das perguntas selecionadas, representados na linha *Weight*; (*iv*) as respostas com maior percentual de diagnósticos, representadas na linha *Answer*; e, por fim, (*v*) as categorias de Nível 1 associadas às perguntas, representadas na linha *Category*. Os *workflows* de diagnóstico resultantes para os Cenários 1 e 2 são apresentados, respectivamente, nas Figuras 4.3(a) e 4.3(b). As perguntas exibidas com o fundo cinza foram respondidas automaticamente pelo sistema. Isto ocorreu pois o mó-

dulo *Question Verifier* considerou tais perguntas como óbvias, ou seja, a intervenção do operador não foi necessária. Além disso, a linha *Operator's Answer* representa a resposta utilizada no diagnóstico, seja informada pelo operador ou atribuída automaticamente pelo sistema.

É importante enfatizar que cada resposta informada influencia diretamente na seleção da próxima pergunta. Isso 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 operador responder às perguntas enviadas, a solução proposta identificou corretamente duas RCs. Para o Cenário 1 foi diagnosticada a RC *O path dos arquivos de e-mail está errado*. Enquanto que para o Cenário 2 foi diagnosticada a RC *As portas do Switch não foram configuradas corretamente*.

Um aspecto que deve ser salientado, refere-se às duas RCs que foram identificadas a partir da mesma atividade que falhou, porém, em cenários com infraestruturas distintas. Isso permite evidenciar a adaptabilidade da solução às nuances comumente encontradas nos ambientes de TI. Além disso, é importante notar que os *workflows* gerados para cada cenário são diferentes. Isso se deve ao fato da infraestrutura sobre a qual o CP foi implementado ser diferente nos dois cenários, possibilitando a ocorrência de diversas falhas. Adicionalmente, é importante destacar que a solução desenvolvida 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 Sistema de Diagnóstico após a sua documentação/especificação por um operador.

4.2 Segundo Estudo de Caso

Nesse segundo estudo de caso, a solução foi aplicada sobre um cenário que simula a instanciação de uma infraestrutura de TI real. Além disso, as quatro estratégias propostas, detalhadas na Seção 3.4, foram aplicadas. Dessa forma, é possível analisar os resultados gerados observando eventuais tendências. É importante salientar que na atividade que falhou, a infraestrutura e a RC identificada não são alteradas ao decorrer desse estudo de caso. Isso permite comparar e avaliar os *workflows* de diagnósticos resultantes, bem como demonstrar o correto funcionamento da solução considerando um ambiente corporativo real, destacando as demais características da solução, entre elas, a flexibilidade, a interação, o reúso do conhecimento e a compatibilidade com os padrões atualmente utilizados.

O cenário analisado é baseado em uma empresa que atua na prestação de serviços Web, tais como, hospedagem de *sites* e envio/recebimento de *e-mails*. Ademais, tal empresa possui um *site* onde disponibiliza a venda dos seus serviços de forma *on-line*, bem como fornece acesso ao serviço de *webmail*. Dessa forma, a atual infraestrutura da empresa consiste em dois servidores denominados *DB Server* e *Web Server*. No *DB Server* estão hospedados os CIs associados apenas ao serviço de banco de dados, tais como, *MySQL* e *DataBase Access*. Por outro lado, o *Web Server* é utilizado para propósito geral e hospeda os demais serviços da empresa, tais como, *site* para vendas *on-line*, serviço de troca de *e-mails* e hospedagem de *sites* de clientes.

Entretanto, visando a atender a crescente demanda de clientes, bem como melhorar os serviços oferecidos, faz-se necessária uma atualização na infraestrutura de TI. Para tanto, a empresa optou por implantar outros dois servidores e, consequentemente, migrar alguns serviços, otimizando o desempenho e incrementando a disponibilidade de recursos. No primeiro servidor a ser implantando, denominado *Hosting Server*, serão hospedados os

CIs relativos ao serviço de hospedagem de *sites*. Por outro lado, no segundo servidor, denominado *Mail Server*, serão hospedados os CIs relativos aos serviços de filtro e troca de *e-mails*. A fim de realizar essas alterações, uma RFC que permite implantar os servidores e posteriormente migrar os serviços selecionados foi criada.

Conforme detalhado anteriormente, a RFC criada é submetida a um processo de refinamento, realizado pelo sistema CHANGELEDGE. Tal processo resulta na criação de um CP, representado na Figura 4.4. Como pode ser observado, as atividades do CP estão divididas sobre qual sistema computacional serão executadas. As atividades com o fundo branco serão executadas sobre o *Web Server*. Por outro lado, as atividades com o fundo cinza claro serão executadas sobre o *Mail Server*. De forma semelhante, as atividades com o fundo cinza escuro serão executadas sobre o *Hosting Server*.

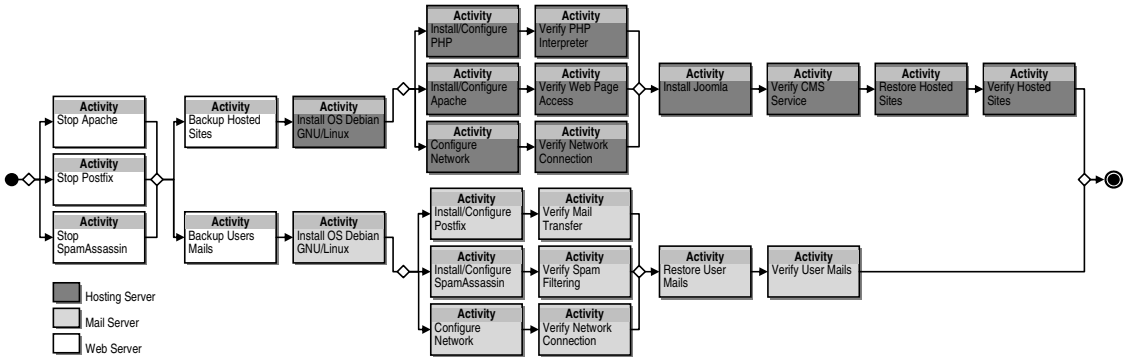


Figura 4.4: *Change Plan* para a implantação de novos servidores e migração de serviços

Inicialmente, os serviços são parados a fim de evitarem que arquivos sejam corrompidos, bem como evitarem que algum dado não seja contemplado pelo processo de *backup* que será realizado. Logo após, são gerados *backups* tanto dos *sites* hospedados, quanto dos *e-mails* dos usuários. Posteriormente, as atividades de instalação e configuração dos *softwares* Debian GNU/Linux, PHP, Apache e Joomla são executadas sobre o *Hosting Server*, bem como a configuração e verificação da conexão de rede. Da mesma forma, as atividades de instalação e configuração dos *softwares* Debian GNU/Linux, Postfix e SpamAssassin são executadas sobre o *Mail Server*, bem como a configuração e verificação da conexão de rede. Por fim, as atividades de restauração e verificação dos sites hospedados e dos *e-mails* dos usuários, são executadas, respectivamente, sobre o *Hosting Server* e *Mail Server*.

Ao executar o CP detalhado anteriormente, o Sistema de Implantação detecta uma falha na atividade *Verify Hosted Sites*. A infraestrutura de TI da empresa nesse instante é detalhada na Figura 4.5. Os diferentes CIs mapeados pelo CIM são representados por retângulos, sendo que os retângulos: (*i*) em cinza claro representam serviços mapeados pela classe *Service*; (*ii*) em cinza representam *softwares* mapeados pela classe *Software Element*; (*iii*) em cinza escuro representam sistemas computacionais mapeados pela classe *Computer System*; e, (*iv*) em branco representam dispositivos de rede mapeados pela classe *Network Device*. As dependências entre os elementos são mapeadas pelas setas. As setas contínuas mapeiam as dependências diretas, enquanto que as setas tracejadas mapeiam as dependências de serviços hospedados em outros sistemas computacionais e, portanto, necessitam de uma conexão de rede para prover a comunicação.

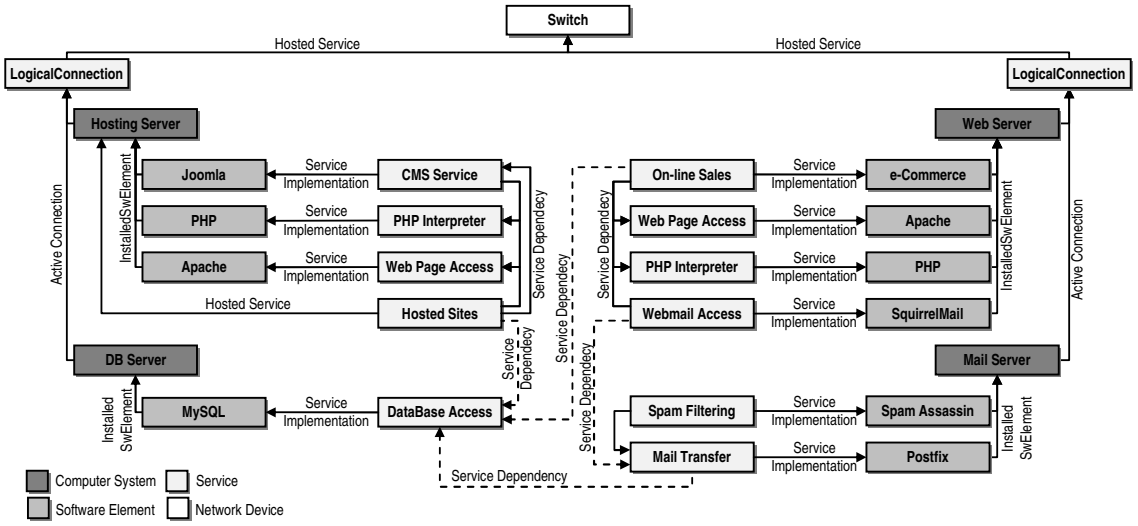


Figura 4.5: Representação da infraestrutura da empresa quando a falha foi reportada

É importante salientar que na Figura 4.5 são representados todos os CIs da infraestrutura TI. Assim sendo, quando um PR é gerado, o sistema identifica através das dependências a infraestrutura parcial que pode ter influenciado ou originado a falha. Como mencionado anteriormente, a atividade que falhou foi *Verify Hosted Sites*. Tal atividade é executada no CI *Hosted Sites*. Dessa forma, os CIs dependentes selecionados são: (*i*) os serviços, *Database Access*, *Web Page Access*, *PHP Interpreter*, *CMS Service* e *Logical Connection*; (*ii*) os *softwares*, Joomla, PHP, Apache e MySQL; (*iii*) os sistemas computacionais, *DB Server* e *Hosting Server*; e, por fim, (*iv*) o dispositivo de rede, *Switch*.

Tabela 4.3: CIs identificados e categorias associadas

CI	**Categorias Associadas**
Hosted Sites	Service ⇒ Web Page Server
DataBase Access	Service ⇒ DataBase
Web Page Access	Service ⇒ Web Page Server
PHP Interpreter	Service ⇒ Web Page Server
CMS Service	Service ⇒ Web Page Server
Logical Connection	Service ⇒ Web Page Server
Joomla	Software ⇒ Web Server
PHP	Software ⇒ Web Server
Apache	Software ⇒ Web Server
MySQL	Software ⇒ Web Server
DB Server	System ⇒ Computer System ⇒ DB Server
Hosting Server	System ⇒ Computer System ⇒ Hosting Server
Switch	Network ⇒ Devices

Os CIs identificados pelas dependências, bem como as categorias associadas são apresentados na Tabela 4.3. A fim de melhorar a compreensão e a legibilidade, os CIs idênticos são apresentados uma única vez. Isto pode ser observado na ocorrência do CI *Logical Connection* representado duas vezes na Figura 4.5 e uma única vez na Tabela 4.3. No en-

tanto, o resultado gerado na execução da solução proposta não sofre nenhuma alteração. Pois, apesar de serem instâncias diferentes, os CIs idênticos estão associados às mesmas categorias. Assim, durante a seleção das categorias, a solução irá identificar as mesmas categorias, independentemente de analisar uma ou duas instâncias de *Logical Connection*.

Na Tabela 4.4[1] são apresentadas as categorias identificadas, bem como os pesos calculados, por cada estratégia proposta, para a primeira interação. Como pode ser observado, a categoria *Service* é considerada de Nível 1 e possui duas categorias de Nível 2 associadas, *Web Page Server* e *DataBase*. Além disso, as estratégias propostas adicionam o peso das categorias de nível inferior ao peso da categoria de nível superior associada. Assim sendo, o peso das categorias *Web Page Server* e *DataBase* é adicionado ao peso da categoria *Service*. É importante salientar que a categoria de maior nível é classificada como Nível 1, enquanto que a categoria de menor nível é classificada como Nível 4. Tal característica explica o elevado peso das categorias de Nível 1 em comparação com as demais categorias.

Tabela 4.4: Categorias identificadas e os pesos calculados pelas estratégias propostas

Categoria	**Nível**	**Peso**			
		Est. 1	**Est. 2**	**Est. 3**	**Est. 4**[1]
Service	1	1083	242	157,30	242
Web Page Server	2	558	82	33,20	82
DataBase	2	519	195	127,60	195
Network	1	1058	345	188,10	345
Services	2	512	189	113,40	189
Devices	2	485	136	66,20	136
System	1	603	167	54,30	167
Computer System	2	545	153	52,90	153
Hosting Server	3	319	175	49,90	175
DB Server	3	192	-22	3,00	-22
Software	1	1115	343	126,60	343
Web Server	2	607	138	86,80	138
DB Server	2	443	169	36,20	169

A fim de selecionar uma pergunta, o sistema identifica qual a categoria de Nível 1 possui o maior peso. Como pode ser observado na Tabela 4.4, a Estratégia 1 seleciona a categoria *Software*. Por outro lado, as demais estratégias selecionam a categoria *Network*. Apesar de identificarem a mesma categoria, as Estratégias 2, 3 e 4 podem resultar na seleção de perguntas diferentes. Isto se deve ao fato das estratégias utilizarem métricas diferentes para calcular o peso das perguntas. Além disso, ao identificar uma categoria de Nível 1, as perguntas que participam da seleção podem estar associadas a qualquer categoria dependente. Assim, ao utilizar a Estratégia 1, as perguntas que participam da seleção, durante a primeira interação, devem estar associadas às categorias *Software*, *Web Server* ou *DB Server*. Semelhantemente, ao utilizar a Estratégia 2, 3 ou 4, as perguntas que participam da seleção, durante a primeira interação, devem estar associadas às categorias *Network*, *Services* ou *Devices*.

[1]Os pesos calculados pela Estratégia 4 e associados às categorias, são idênticos aos pesos calculados pela Estratégia 2. Isto se deve ao fato de que as duas estratégias são idênticas no que tange ao cálculo dos pesos das RCs e categorias. Porém, as estratégias diferenciam quanto ao cálculo do peso das perguntas, onde a Estratégia 4 considera a popularidade das perguntas.

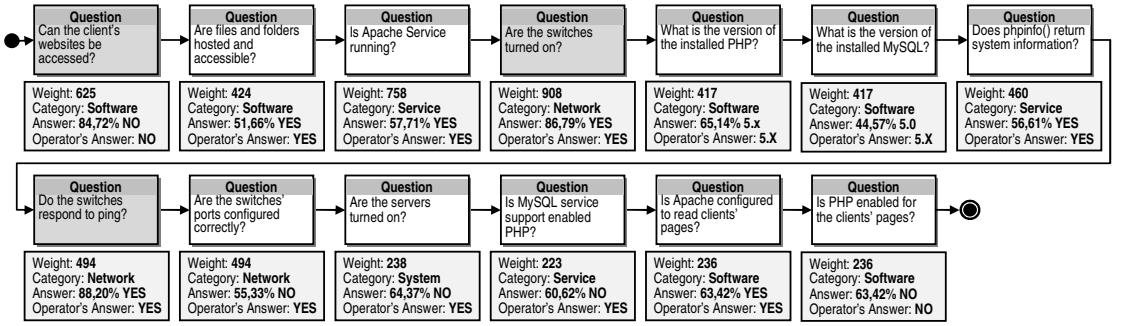
Uma observação mais detalhada é necessária no que tange aos pesos calculados pelas estratégias propostas. Como a Estratégia 1 não considera os diagnósticos frustrados, os pesos calculados são elevados em relação às demais estratégias. Isso permite a seleção de perguntas associadas às RCs que possuam uma grande quantidade de diagnósticos concluídos, porém, com um número também elevado de diagnósticos frustrados. Além disso, essa estratégia posterga a identificação de RCs recentemente adicionadas. Isto se deve à discrepância entre os pesos de uma RC recentemente adicionada e uma RC identificada diversas vezes ao longo do tempo. Com o intuito de reduzir tal discrepância, a Estratégia 2 penaliza as RCs por cada diagnóstico frustrado associado.

Apesar de amenizar a discrepância entre os pesos, a Estratégia 2 não considera a idade dos diagnósticos. Dessa forma, RCs que foram identificadas como a causa de um grande número de falhas em um passado distante, têm suas perguntas selecionadas em diversas interações até que seu peso seja reduzido. Isto gera um grande número de interações adicionais e, consequentemente, desperdiça o tempo do operador. Assim sendo, a Estratégia 3 considera a idade dos *logs* e assim prioriza o histórico de diagnósticos recentes. Como pode ser observado na Tabela 4.4, ao se comparar os pesos calculados para a categoria *Software*, é possível observar uma drástica redução entre os valores obtidos pelas Estratégias 2 e 3. Tal redução é consequência da penalização gerada com base na idade dos *logs*. Por conseguinte, a probabilidade de seleção da categoria *Software* é minimizada, resultando na seleção de categorias com histórico de identificações mais recente.

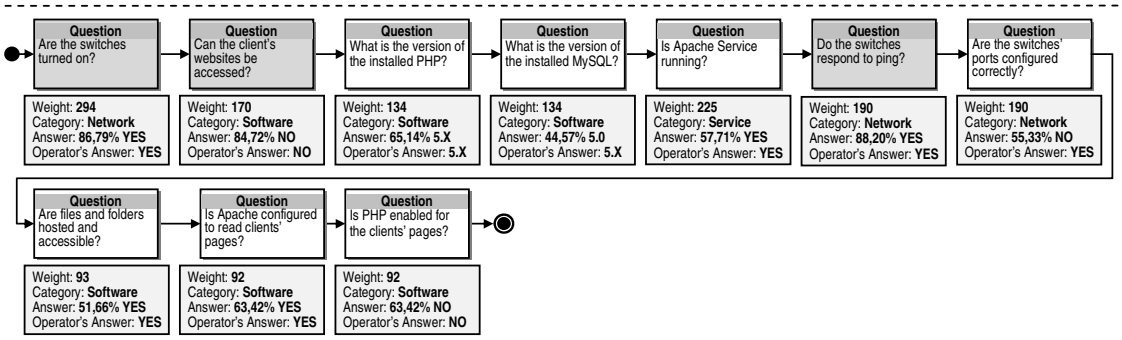
Os *workflows* de diagnóstico gerados pela solução, após a execução dos passos detalhados no Capítulo 3, são representados na Figura 4.6. O *workflow* gerado pela Estratégia 1 é representado na Figura 4.6(a). Do mesmo modo, os *workflows* de diagnóstico apresentados na Figura 4.6(b), 4.6(c) e 4.6(d) representam os resultados gerados utilizando, respectivamente, as Estratégias 2, 3 e 4. Os retângulos com fundo cinza representam as perguntas que foram respondidas pelo módulo *Question Verifier*, enquanto que os retângulos com fundo branco representam as perguntas respondidas pelo operador. Abaixo das perguntas, uma caixa contendo informações sobre a seleção da pergunta é exibida. Entre as informações apresentadas estão: (*i*) o peso da pergunta, informado na linha *Weight*; (*ii*) a resposta com o maior percentual de diagnósticos, informado na linha *Answer*; (*iii*) a resposta utilizada, informada na linha *Operator's Answer*; e, (*iv*) a categoria Nível 1 selecionada, informada na linha *Category*. Além disso, apenas para a Estratégia 4 é apresentada também a popularidade da pergunta selecionada. Tal informação é representada na linha *Popularity*.

Como pode ser observado na Figura 4.6, a Estratégia 4 identificou a RC com o menor número de interações. Por outro lado, na utilização da Estratégia 1 foi observado o maior número de interações. A redução no número de interações, observada na comparação entre as estratégias, é resultado da utilização de diferentes métricas para o cálculo dos pesos. Enquanto que a Estratégia 1 considera apenas os diagnósticos concluídos, a Estratégia 4 considera todo o histórico de diagnósticos, bem como a popularidade de cada pergunta. Isto permite selecionar primeiro as questões que estão associadas a diversas RCs, eliminando RCs que possuam perguntas em comum, e que não possam ser identificadas, com uma menor quantidade de interações.

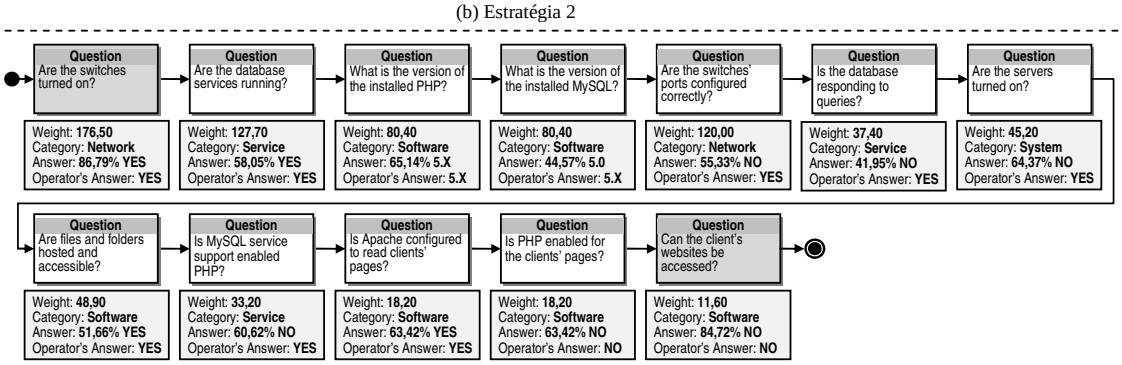
De certa forma, as estratégias propostas já priorizam perguntas associadas à diversas RCs, pois adicionam, ao peso da pergunta, os peso de todas as RCs associadas. Assim, perguntas constantes nos conjuntos de diagnóstico de várias RCs possuem um peso maior que as perguntas constantes no conjunto de diagnóstico de apenas uma dessas RCs. Porém, há casos em que duas perguntas possuem o mesmo peso ou pesos aproximados.



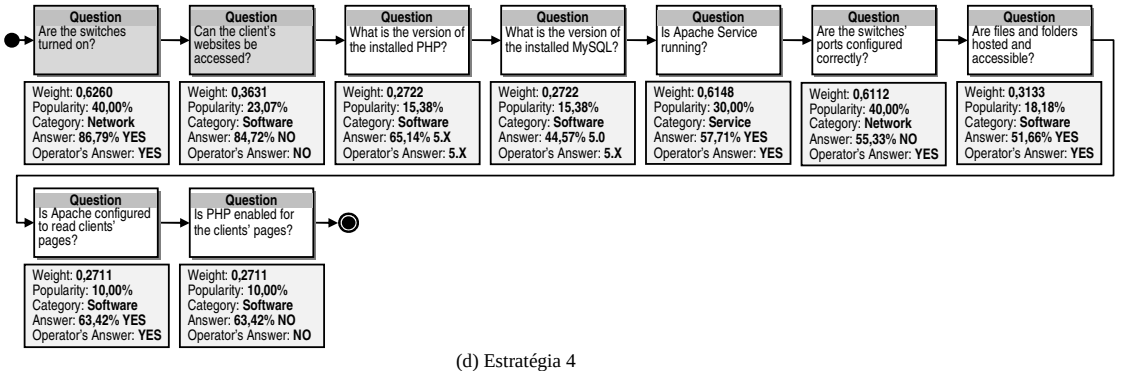
(a) Estratégia 1



(b) Estratégia 2



(c) Estratégia 3



(d) Estratégia 4

Figura 4.6: Workflows de diagnóstico gerados utilizando diferentes estratégias

Considerando esses casos, a Estratégia 4 é a única que prioriza as perguntas constantes em diversos conjuntos de diagnóstico, pois incrementa o peso das perguntas mais populares. Isso pode ser observado no *workflow* de diagnóstico da Estratégia 4, que selecionou a pergunta "*Are the switches' ports configured correctly?*" em detrimento da pergunta "*Do the switches respond to ping?*", selecionada pela Estratégia 2. É importante lembrar que as Estratégias 2 e 4 utilizam a mesma abordagem para a seleção das categorias, porém métricas diferentes para a seleção das perguntas.

Um aspecto que deve ser observado, refere-se à quantidade de categorias de Nível 1 selecionadas. As Estratégias 1 e 3 alteram 9 vezes a categoria utilizada para a seleção das perguntas. Por outro lado, as Estratégias 2 e 4 alteram apenas 5 vezes a categoria utilizada. Essa quantidade de alterações reflete no número de interações entre o sistema e o operador, pois a cada alteração de categoria, no mínimo uma pergunta é selecionada. Aliado a isto, a pergunta "*Does phpinfo() return system information?*" é selecionada apenas pela Estratégia 1. Semelhantemente, as perguntas "*Are the database services running?*" e "*Is the database responding to queries?*" são selecionadas apenas pela Estratégia 3. Isso ocorre por dois principais motivos: (*i*) o elevado número de diagnósticos considerados antigos e penalizados na Estratégia 3, resulta em um nivelamento dos pesos e acarreta na seleção de perguntas não necessárias em outras estratégias; e, (*ii*) a grande quantidade de diagnósticos frustrados não penalizados pela Estratégia 1, resulta em pesos elevados para RCs que possuam um histórico heterogêneo e, consequentemente, prioriza perguntas que não refletem corretamente as experiências passadas. Assim, a utilização dessas métricas pelas Estratégias 1 e 3 acarretam na seleção de perguntas específicas e em uma maior quantidade de alterações de categorias Nível 1 selecionadas.

Uma outra observação se faz necessária sobre a pergunta "*Can the client's websites be accessed?*". Nas Estratégias 1, 2 e 4 essa pergunta é uma das primeiras selecionadas. Porém, ao utilizar a Estratégia 3 tal pergunta é selecionada somente no final do processo de diagnóstico. Isso se deve ao fato de que existem diversos diagnósticos associados que sofrem penalizações pela sua idade, bem como, há uma grande quantidade de diagnósticos frustrados. Assim sendo, o peso da pergunta é drasticamente reduzido, resultando na seleção de tal pergunta somente ao final do processo de identificação.

É importante salientar que apesar de considerar a mesma infraestrutura de TI e a mesma falha reportada, as quatro estratégias propostas geraram diferentes *workflows* de diagnóstico. Tais *workflows* diferem-se não apenas na quantidade de perguntas, mas também, nas perguntas pelas quais são compostos. Apesar disso, os diagnósticos gerados resultaram na identificação de uma única RC. Após o operador responder às perguntas, ao utilizar qualquer uma das estratégias desenvolvidas, foi identificada corretamente a RC: "*A configuração do PHP não permite a utilização da linguagem em sites de usuários*".

5 CONSIDERAÇÕES FINAIS 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 auxiliarem nessa etapa não utilizam métodos que possam reaproveitar facilmente o conhecimento adquirido com experiências anteriores. Consequentemente, falhas na execução de mudanças podem ocorrer indefinidamente sem que os operadores saibam exatamente o que está causando tais falhas. Com o intuito de abordar temas relacionados a esse problema, diversas pesquisas foram desenvolvidas pela comunidade científica, como explanado durante o Capítulo 2.

Embora a área de ITSM tenha recebido grande atenção da comunidade científica em investigações recentes, nenhuma das pesquisas detalhadas anteriormente permite a identificação da causa raiz de falhas, ocorridas na execução de mudanças. Ademais, poucos trabalhos exploram a possibilidade de aprendizagem a partir de experiências anteriores. Para tratar tais limitações, ao longo desta dissertação foi proposta uma solução conceitual, juntamente com uma expansão do modelo CIM, que permite a identificação da causa raiz de falhas em execuções de mudanças de TI. É importante salientar que a solução proposta está situada na área de Gerenciamento de Mudanças, no domínio reativo, em que diferentes fontes de falhas são consideradas que não possibilitam diagnosticá-las de forma automática, por exemplo, erros humanos.

5.1 Principais Contribuições e Resultados Obtidos

A principal contribuição desta dissertação é a solução proposta para a identificação da causa raiz de falhas, ocorridas em mudanças de TI. Como mencionado anteriormente, em contraste com as soluções existentes, a solução proposta permite reutilizar tanto os traços históricos de diagnósticos, quanto a experiência do operador. Isso possibilita otimizar o processo de identificação da causa raiz de falhas e, consequentemente, reduzir o impacto, os custos atrelados e o tempo despendido nesse processo. Outrossim, a estrutura modular da solução desenvolvida permite dividir a complexidade do processo de identificação em problemas menores que podem ser resolvidos individualmente. Além disso, a utilização de módulos permite que as organizações personalizem partes da solução para melhor refletirem as necessidades especiais do seu ambiente de TI.

Outras contribuições também devem ser mencionadas. Primeiro, a expansão do modelo CIM, detalhada na Seção 3.1, que permite representar informações sobre diagnósticos anteriores e causas raiz identificadas. Tal modelo foi utilizado como referência por ser amplamente utilizado pelas organizações para descrever, de forma detalhada, os elementos da infraestrutura de TI, tais como, sistemas computacionais, pessoas e processos, bem como as relações entre esses elementos. Segundo, a adaptabilidade do diagnóstico tanto

a infraestrutura de TI, quanto a experiência do operador. Isso permite considerar, durante o diagnóstico, a infraestrutura que é afetada e/ou que pode ser a responsável pela falha relatada. Além disso, a interação entre o operador e o sistema permite identificar falhas de difícil diagnóstico automático, tais como, erros humanos, falhas em dispositivos de conexão e erros de configuração. Por fim, as estratégias propostas para a seleção da pergunta que deve ser enviada ao operador. Essas estratégias permitem priorizar perguntas com determinadas características, tais como, perguntas mais vezes utilizadas, perguntas populares e perguntas com um maior histórico recente.

Os resultados obtidos, apesar de não exaustivos, evidenciam os benefícios de utilizar a solução proposta para identificar as causas raiz de falhas. No primeiro estudo de caso, detalhado na Seção 4.1, é demonstrada a adaptabilidade do diagnóstico, gerado pela solução, tanto à infraestrutura de TI onde a falha ocorre, quanto às respostas fornecidas pelo operador. A partir de uma mesma falha, a solução gerou diferentes *workflows* de diagnóstico para cada cenário analisado. Isto ocorre, pois os conjuntos de CIs previamente identificados, em cada cenário, são distintos e acarretam na seleção de diferentes instâncias de categorias, *logs* e conjuntos de diagnóstico. Com base nas categorias selecionadas, é possível identificar diversas causas raiz a partir de uma mesma falha relatada. Isso permite evidenciar a adaptabilidade da solução às nuances comumente encontradas nos ambientes de TI.

No segundo estudo de caso, discutido na Seção 4.2, as diferenças entre as quatro estratégias propostas são evidenciadas, bem como é demonstrado o correto funcionamento da solução considerando um ambiente corporativo real. Como apresentado anteriormente, cada estratégia gerou um *workflow* de diagnóstico diferente considerando a mesma falha, bem como a mesma infraestrutura de TI. Pôde-se observar, ainda, que os *workflows* resultantes diferem-se tanto na quantidade de perguntas, quanto nas perguntas pelas quais são compostos. Apesar disto, as estratégias aplicadas identificaram corretamente a causa raiz. Assim, não é possível afirmar com 100% de certeza qual estratégia é melhor, visto que os *workflows* gerados dependem das características do histórico de diagnósticos. Como resultado de ambos os estudos de caso, foi observada a identificação das causas raiz de falhas ocorridas durante a execução de mudanças, assim como as demais características da solução, entre elas, a flexibilidade, a interação, o reúso do conhecimento e a compatibilidade com os padrões atualmente utilizados.

Com base nas características do histórico de diagnósticos, é possível recomendar o uso das estratégias desenvolvidas. A Estratégia 1 é recomendada para uso com históricos em formação. Nesses casos é importante não aplicar penalizações, pois resultariam na aproximação demasiada dos pesos das perguntas e aumentariam o número de interações do processo de diagnóstico. A utilização da Estratégia 2 é recomendada para históricos volumosos e de pouca idade. Assim, as perguntas são penalizadas por cada diagnóstico frustrado. Tal penalização prioriza as perguntas de acordo com o histórico, bem como permite que perguntas associadas às novas RCs sejam utilizadas rapidamente. Por outro lado, a utilização da Estratégia 3 é recomendada para *logs* que contemplem pelo menos 10 meses de experiências passadas. Isso permite utilizar-se a métrica idade somente quando se for realmente otimizar os resultados obtidos, evitando computações desnecessárias. Ademais, a Estratégia 4 é recomendada para bases de dados com uma grande quantidade de perguntas populares. Isso se deve ao fato de tal estratégia otimizar a seleção dessas perguntas, reduzindo o número de interações. De maneira geral, as Estratégias 2 e 4 demonstraram melhores resultados considerando cenários com dados históricos aleatórios.

A partir das investigações realizadas e dos resultados obtidos, o trabalho descrito nesta dissertação foi apresentado e discutido com a comunidade em eventos científicos nacionais e internacionais de grande relevância. As críticas, de modo geral, foram extremamente positivas, o que demonstra a importância e a qualidade do trabalho desenvolvido. Algumas das sugestões feitas pelos revisores estão consolidadas nesta dissertação. Abaixo segue a relação dos dois artigos aprovados dentro do escopo deste trabalho:

- "A Solution for Identifying the Root Cause of Problems in IT Change Management". **Mini-conference of 12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011)**, Dublin, Irlanda. Proceedings of the 12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011). New York, NY: IEEE Communications Society, 2011. p. 592-599. (SANTOS et al., 2011);

- "Identificação Interativa da Causa Raiz de Problemas em Execuções de Mudanças de TI". **XXIX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2011)**, Campo Grande, MS. Anais do XXIX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos. Porto Alegre, RS: Sociedade Brasileira de Computação, 2011. p. 891-904. (SANTOS et al., 2011).

5.2 Questões em Aberto e Possíveis Investigações Futuras

Apesar dos progressos alcançados, existem diversas possíveis melhorias e investigações futuras que merecem ser mencionadas. Considerando investigações que podem ser realizadas em curto prazo, sugere-se o desenvolvimento de novas estratégias para o módulo *Question Selector*. Uma oportunidade vislumbrada refere-se à avaliação de outras características do histórico de diagnósticos, tais como, taxa de falsos positivos e falsos negativos. Isto permitiria criar uma nova métrica, a confiança dos diagnósticos. Através dessa métrica seria possível diferenciar perguntas que possuem pesos iguais, porém, históricos diferentes. Por exemplo, P_1 possui 25 diagnósticos concluídos e apenas 5 diagnósticos frustrados. Por outro lado, P_2 possui 40 diagnósticos concluídos e 20 diagnósticos frustrados. Assim, P_1 e P_2 possuem o mesmo peso calculado, valor igual a 20. Através da utilização da confiança seria possível diferenciar P_1 e P_2, priorizando perguntas com uma menor quantidade de diagnósticos frustrados. Outra oportunidade refere-se à extensão do processo de identificação de causas raiz para outros escopos, por exemplo, aplicando os métodos propostos para gerenciar incidentes em operações/suporte de serviços.

Como investigação futura, em longo prazo, sugere-se a avaliação do uso de classes mapeadas pelo modelo CIM, tais como, dependências, *checks* e *actions*, a fim de identificar os requisitos dos CIs para seu correto funcionamento. Dessa forma, ao iniciar o processo de diagnóstico, seria possível detectar se os CIs dependentes apresentam alguma falha e, assim, otimizar o processo de identificação da causa raiz. Ainda mais, essa melhoria permitiria otimizar o *bootstrapping* do sistema, possibilitando identificar falhas mesmo quando não há histórico de diagnósticos. Um outro aspecto que é sugerido para investigação é automatizar a identificação da causa raiz de determinadas falhas. Dessa maneira, falhas que podem ser identificadas automaticamente poderiam ser diagnosticadas sem a intervenção humana, tais como, ausência de bibliotecas durante a instalação de um *software* e erros de conexão com determinados dispositivos. É importante mencionar que a maioria dessas falhas está relacionada à *softwares* e que na literatura existem propostas que permitem tal automatização, conforme mencionado no Capítulo 2.

REFERÊNCIAS

AGARWAL, M.; MADDURI, V. Correlating failures with asynchronous changes for root cause analysis in enterprise environments. In: IEEE/IFIP INTERNATIONAL CONFERENCE ON DEPENDABLE SYSTEMS AND NETWORKS (DSN), 2010, Chicago, EUA. **Proceedings...** Piscataway: IEEE Operations Center, 2010. p.517 –526.

APPLEBY, K.; GOLDSZMIDT, G.; STEINDER, M. Yemanja-a layered event correlation engine for multi-domain server farms. In: IFIP/IEEE INTERNATIONAL SYMPOSIUM ON INTEGRATED NETWORK MANAGEMENT (IM), 2001, Seattle, EUA. **Proceedings...** Piscataway: IEEE Operations Center, 2001. p.329 –344.

BARTOLINI, C.; SALLE, M.; TRASTOUR, D. IT service management driven by business objectives An application to incident management. In: IEEE/IFIP NETWORK OPERATIONS AND MANAGEMENT SYMPOSIUM (NOMS), 2006, Vancouver, Canadá. **Proceedings...** Piscataway: IEEE Operations Center, 2006. p.45–55.

BARTOLINI, C.; STEFANELLI, C. Business-driven IT management. In: IFIP/IEEE INTERNATIONAL SYMPOSIUM ON INTEGRATED NETWORK MANAGEMENT (IM), 2011, Dublin, Irlanda. **Proceedings...** Piscataway: IEEE Operations Center, 2011. p.964 –969.

BIANCHIN, L.; WICKBOLDT, J.; GRANVILLE, L.; GASPARY, L.; BARTOLINI, C.; RAHMOUNI, M. Similarity metric for risk assessment in IT change plans. In: IEEE/IFIP INTERNATIONAL CONFERENCE ON NETWORK AND SERVICE MANAGEMENT (CNSM), 2010, Niagara Falls, Canadá. **Proceedings...** Piscataway: IEEE Operations Center, 2010. p.25 –32.

BROWN, A.; KELLER, A. A Best Practice Approach for Automating IT Management Processes. In: IEEE/IFIP NETWORK OPERATIONS AND MANAGEMENT SYMPOSIUM (NOMS), 2006, Vancouver, Canadá. **Proceedings...** Piscataway: IEEE Operations Center, 2006. p.33 –44.

CORDEIRO, W. L. C.; MACHADO, G. S.; ANDREIS, F. G.; SANTOS, A. D.; BOTH, C. B.; GASPARY, L. P.; GRANVILLE, L. Z.; BARTOLINI, C.; TRASTOUR, D. ChangeLedge: Change Design and Planning in Networked Systems based on Reuse of Knowledge and Automation. **Computer Networks**, New York, v.53, n.16, p.2782–2799, Nov. 2009.

DIAO, Y.; JAMJOOM, H.; LOEWENSTERN, D. Rule-Based Problem Classification in IT Service Management. In: IEEE INTERNATIONAL CONFERENCE ON CLOUD

COMPUTING (CLOUD), 2009, Bangalore, Índia. **Proceedings...** Piscataway: IEEE Operations Center, 2009. p.221 –228.

DMTF. **CIM - Common Information Model**. Disponível em: <http://www.dmtf.org/standards/cim>. Acesso em: Nov. 2009.

GUPTA, R.; PRASAD, K.; MOHANIA, M. Automating ITSM Incident Management Process. In: IEEE INTERNATIONAL CONFERENCE ON AUTONOMIC COMPUTING (ICAC), 2008, Chicago, EUA. **Proceedings...** Piscataway: IEEE Operations Center, 2008. p.141 –150.

HAGEN, S.; KEMPER, A. Towards solid IT Change Management: automated detection of conflicting it change plans. In: IFIP/IEEE INTERNATIONAL SYMPOSIUM ON INTEGRATED NETWORK MANAGEMENT (IM), 2011, Dublin, Irlanda. **Proceedings...** Piscataway: IEEE Operations Center, 2011. p.265 –272.

ISACA. **Control Objectives for Information and related Technologies (COBIT)**. Disponível em: <http://www.isaca.org/cobitonline/>. Acesso em: Jan. 2011.

ISO. **ISO 20000**:2005 Information technology – Service Management. Geneva, Suíca: International Organization for Standardization, 2005.

JANTTI, M.; EEROLA, A. A Conceptual Model of IT Service Problem Management. In: IEEE INTERNATIONAL CONFERENCE ON SERVICE SYSTEMS AND SERVICE MANAGEMENT (ICSSSM), 2006, Troyes, França. **Proceedings...** Piscataway: IEEE Operations Center, 2006. p.798–803.

KELLER, A. Automating the Change Management Process with Electronic Contracts. In: IEEE INTERNATIONAL CONFERENCE ON E-COMMERCE TECHNOLOGY WORKSHOPS (CECW), 2005, Washington, EUA. **Proceedings...** Piscataway: IEEE Operations Center, 2005. p.99–108.

KELLER, A.; HELLERSTEIN, J.; WOLF, J.; WU, K.; KRISHNAN, V. The CHAMPS system: Change management with planning and scheduling. In: IEEE/IFIP NETWORK OPERATIONS AND MANAGEMENT SYMPOSIUM (NOMS), 2004, Seoul, Coréia do Sul. **Proceedings...** Piscataway: IEEE Operations Center, 2004. p.395–408.

LUNARDI, R.; ANDREIS, F.; COSTA CORDEIRO, W. da; WICKBOLDT, J.; DALMAZO, B.; SANTOS, R. dos; BIANCHIN, L.; GASPARY, L.; GRANVILLE, L.; BARTOLINI, C. On strategies for planning the assignment of human resources to IT change activities. In: IEEE/IFIP NETWORK OPERATIONS AND MANAGEMENT SYMPOSIUM (NOMS), 2010, Osaka, Japão. **Proceedings...** Piscataway: IEEE Operations Center, 2010. p.248 –255.

LUNARDI, R. C.; CORDEIRO, W. L. C.; ANDREIS, F. G. et al. ChangeAdvisor: A Solution to Support Alignment of IT Change Design with Business Objectives/Constraints. In: IFIP/IEEE INTERNATIONAL WORKSHOP ON DISTRIBUTED SYSTEMS: OPERATIONS AND MANAGEMENT (DSOM), 2009, Venice, Itália. **Proceedings...** Piscataway: IEEE Operations Center, 2009. p.138–151.

MACHADO, G. S.; CORDEIRO, W. L. C.; DAITX, F. F.; BOTH, C. B.; GASPARY, L. P.; GRANVILLE, L. Z.; BARTOLINI, C.; SAHAI, A.; TRASTOUR, D.; SAIKOSKI, K. Enabling Rollback Support in IT Change Management Systems. In: IEEE/IFIP NETWORK OPERATIONS AND MANAGEMENT SYMPOSIUM (NOMS), 2008, Salvador, Brasil. **Proceedings...** Piscataway: IEEE Operations Center, 2008. p.347–354.

MACHADO, G. S.; CORDEIRO, W. L. C.; SANTOS, A. D.; WICKBOLDT, J. A.; ROBEN CASTAGNA LUNARDI, F. G. A.; BOTH, C. B.; GASPARY, L. P.; GRANVILLE, L. Z.; TRASTOUR, D.; BARTOLINI, C. Refined Failure Remediation in IT Change Management Systems. In: IFIP/IEEE INTERNATIONAL SYMPOSIUM ON INTEGRATED NETWORK MANAGEMENT (MINI-CONFERENCE OF IM), 2009, New York, EUA. **Proceedings...** Piscataway: IEEE Operations Center, 2009. p.638–645.

MOURA, A.; SAUVE, J.; BARTOLINI, C. Business-driven IT management - upping the ante of IT : exploring the linkage between it and business to improve both it and business results. **IEEE Communications Magazine**, [S.l.], v.46, n.10, p.148 –153, Out. 2008.

OGC. **Information Technology Infrastructure Library**: Service Transition Version 3.0. London, Inglaterra: Office of Government Commerce, 2007.

OGC. **Information Technology Infrastructure Library**: Service Operation Version 3.0. London, Inglaterra: Office of Government Commerce, 2007.

OGC. **Information Technology Infrastructure Library (ITIL)**. Disponível em: <http://www.itil-officialsite.com/>. Acesso em: Jan. 2011.

PMI. **Project Management Body of Knowledge (PMBOK)**. Disponível em: <http://www.pmi.org/>. Acesso em: Jan. 2011.

RAHMOUNI, M.; BARTOLINI, C. Learning from past experiences to enhance decision support in IT change management. In: IEEE/IFIP NETWORK OPERATIONS AND MANAGEMENT SYMPOSIUM (NOMS), 2010, Osaka, Japão. **Proceedings...** Piscataway: IEEE Operations Center, 2010. p.408 –415.

REBOUÇAS, R.; SAUVÉ, J.; MOURA, A.; BARTOLINI, C.; TRASTOUR, D. A Decision Support Tool to Optimize Scheduling of IT Changes. In: IFIP/IEEE INTERNATIONAL SYMPOSIUM ON INTEGRATED NETWORK MANAGEMENT (IM), 2007, Munich, Alemanha. **Proceedings...** Piscataway: IEEE Operations Center, 2007. p.343–352.

SANTOS, R. dos; WICKBOLDT, J.; LUNARDI, R.; DALMAZO, B.; GASPARY, L.; GRANVILLE, L. Identificação Interativa da Causa Raiz de Problemas em Execuções de Mudanças de TI. In: SIMPóSIO BRASILEIRO DE REDES DE COMPUTADORES E SISTEMAS DISTRIBUíDOS (SBRC), 2011, Campo Grande, Brasil. **Anais...** Porto Alegre: Sociedade Brasileira de Computação, 2011. p.891–904.

SANTOS, R. dos; WICKBOLDT, J.; LUNARDI, R.; DALMAZO, B.; GRANVILLE, L.; GASPARY, L.; BARTOLINI, C.; HICKEY, M. A solution for identifying the root cause of problems in IT change management. In: IFIP/IEEE INTERNATIONAL SYMPOSIUM ON INTEGRATED NETWORK MANAGEMENT (MINI-CONFERENCE OF IM), 2011, Dublin, Irlanda. **Proceedings...** Piscataway: IEEE Operations Center, 2011. p.586 –593.

SAUVÉ, J.; SANTOS, R.; REBOUCAS, R.; MOURA, A.; BARTOLINI, C. Change Priority Determination in IT Service Management Based on Risk Exposure. **IEEE Transactions on Network and Service Management**, [S.l.], v.5, n.3, p.178 –187, Set. 2008.

SAUVÉ, J.; SANTOS, R. A.; ALMEIDA, R. R.; MOURA, J. A. B. On the Risk Exposure and Priority Determination of Changes in IT Service Management. In: IFIP/IEEE INTERNATIONAL WORKSHOP ON DISTRIBUTED SYSTEMS: OPERATIONS AND MANAGEMENT (DSOM), 2007, San Jose, EUA. **Proceedings...** Piscataway: IEEE Operations Center, 2007. p.147–158.

SHUAI, Z.; ZHIQIANG, Z. Most efficiently allocating resources in IT change management. In: IEEE INTERNATIONAL CONFERENCE ON COMMUNICATION SOFTWARE AND NETWORKS (ICCSN), 2011, Karlsruhe, Alemanha. **Proceedings...** Piscataway: IEEE Operations Center, 2011. p.129 –133.

STEINDER, M.; SETHI, A. S. Probabilistic fault diagnosis in communication systems through incremental hypothesis updating. **Computer Networks**, New York, v.45, n.4, p.537–562, Jul. 2004.

WICKBOLDT, J. A.; BIANCHIN, L. A.; LUNARDI, R. C.; GRANVILLE, L. Z.; GASPARY, L. P.; BARTOLINI, C. A framework for risk assessment based on analysis of historical information of workflow execution in IT systems. **Computer Networks**, New York, v.55, n.13, p.2954 – 2975, Set. 2011.

ANEXO A ARTIGO PUBLICADO – IM 2011

Neste anexo, o artigo intitulado "*A Solution for Identifying the Root Cause of Problems in IT Change Management*" é apresentado. Esta foi a primeira publicação sobre o tema de pesquisa, apresentado nesta dissertação, em eventos científicos renomados. Neste artigo, é detalhada uma extensão do modelo CIM, a fim de representar informações sobre diagnósticos da causa raiz. Além disso, é proposta uma solução para a identificação da causa raiz de problemas em mudanças de TI. Ademais, um estudo de caso é conduzido com o intuito de validar a solução desenvolvida, bem como evidenciar as suas principais características, tais como, adaptabilidade à infraestrutura analisada e o reúso de traços históricos.

- **Título:**
  A Solution for Identifying the Root Cause of Problems in IT Change Management
- **Conferência:**
  Mini Conference of 12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011)
- **URL:**
  `http://www.ieee-im.org/2011/`
- **Data:**
  23-27 de Maio de 2011
- **Local:**
  Trinity College Dublin, Dublin, Irlanda
- **Digital Object Identifier (DOI):**
  `http://dx.doi.org/10.1109/INM.2011.5990563`

12th IFIP/IEEE IM 2011: Mini Conference

A Solution for Identifying the Root Cause of Problems in IT Change Management

Ricardo L. dos Santos*, Juliano A. Wickboldt*,
Roben C. Lunardi*, Bruno L. Dalmazo*,
Lisandro Z. Granville* and Luciano P. Gaspary*
*Institute of Informatics
Federal University of Rio Grande do Sul, Brazil
{rlsantos, jwickboldt, rclunardi,
bldalmazo, granville, paschoal}@inf.ufrgs.br

Claudio Bartolini‡ and Marianne Hickey†
‡Hewlett Packard Laboratories, Palo Alto, USA
†Hewlett Packard Laboratories, Bristol, UK
{claudio.bartolini, marianne.hickey}@hp.com

***Abstract*—The reuse of knowledge acquired by operators to diagnose failures in Information Technology (IT) infrastructures has potential to decrease the recurrence of failures and, consequently, reduce possible losses and maintenance costs. Nevertheless, existing solutions to support failure diagnosis lack of flexibility to adapt to a constantly changing IT environment. As a result, diagnostic is performed in an *ad hoc* and static fashion, which hampers the reuse of knowledge to solve similar failures affecting different elements of an IT infrastructure. To bridge this gap, in this paper we propose an extension of Common Information Model (CIM), supported by a conceptual solution for the identification of the root causes of problems, adaptable to changes in the target infrastructure and applicable to similar failures. Experiments carried out considering typical failures during the deployment of IT changes provide evidence about the efficacy of the proposed solution[1].**

I. Introduction

The Information Technology Service Management (ITSM) has received massive attention in recent years. This is mainly due to the increasing importance of Information Technology (IT) resources and services that support the business of organizations [1]. In this context, guides of best practices, such as Information Technology Infrastructure Library (ITIL) [2], have helped organizations to properly maintain their assets and IT services. Such guides have become vital for organizations that provide large scale and highly dynamic services.

Among the disciplines described in ITIL, IT change management is the one that defines how changes should be planned, scheduled, implemented, and evaluated in complex IT infrastructures. The initial specification of a change is expressed in a document called Request for Change (RFC), that defines what needs to be changed, but not necessarily specifies how the change should be implemented. Based on the specifications of an RFC, an operator generates a Change Plan (CP). This plan is essentially a workflow, composed of activities that, when implemented, will make the IT infrastructure migrate to a new consistent state that reflects the changes requested in RFC [3] [4].

[1]This result was achieved in cooperation with Hewlett-Packard Brasil Ltda. using incentives of Brazilian Informatics Law (Law no 8.2.48 of 1991).

Despite the significant improvements that the adoption of ITIL's best practices can provide for the operation and management of networks and services, the occurrence of failures in IT process cannot be neglected by operators and decision makers. To address potential problems in business processes, ITIL proposes the discipline of Problem Management, that is responsible to define the management lifecycle of IT problems [5]. The main goals of such discipline are: (*i*) preventing the occurrence of IT related problems, (*ii*) eliminating recurring problems, and (*iii*) minimizing the impact of problems to the business continuity. To achieve these goals, the reuse of knowledge and operator experience in relation to IT processes is of importance. Because it allows simplification the procedures for detection the root cause of IT problems and hence reducing the associated costs.

To assist in the process of diagnosing problems, organizations take advantage of tools for analysis and identification of the root cause of problems in networks and services. These tools contain pre-defined steps that help diagnosing and mitigating failures. The use of these diagnostic tools, even when they do not point directly into the root cause of failure, is able to provide important information to the responsible technical staff.

Despite presenting some efficacy, the description of cases in diagnostic tools is performed in *ad hoc* fashion (*e.g.*, through scripts containing questions and answers). Mostly, these cases are static, making it practically impossible to reuse them entirely or partly. That limits the identification of similar problems and therefore decreases the probability of success of the diagnosis process. Moreover, because of the constant evolution of services and IT assets, many cases become outdated or not consistent with the current IT infrastructure.

Aiming to support the identification of root causes of problems, in this paper we present a conceptual solution, supported by an extension of the Common Information Model (CIM) [6] which allows reusing partial or complete cases.

In contrast to existing solutions, our proposal is intended to be both flexible, regarding the constant evolution of IT infrastructures, and adaptable to consider new problems as

978-1-4244-9221-3/11/$26.00 ©2011 IEEE

similar to known ones. The root cause identification process aided by such a solution enables reuse of knowledge from cases that already been completed and still leverages on the experience of operators.

Therefore, we allows the identification of the root cause of problems analyzed in an interactive and dynamic fashion, considering for both the infrastructure affected as well as the responses of the operator. Experiments conducted considering recurrent failures IT changes and have shown the effectiveness of the proposed solution in reducing the time to detect known root causes of problems.

The rest of the paper is organized as follows. In Section II we discusses some of the major research efforts related to failures in IT change management. In Section III we introduce the solution for the diagnosis of problems proposed in the scope of this paper. A case study conducted to evaluate the effectiveness of the proposed solution is described in Section IV, whereas Section V concludes the paper with final remarks and perspectives for future works.

II. Related Work

The area of IT change management has received great attention from researchers in recent years. Improvements have been proposed to change processes in order to make them more efficient, agile, and less expensive. Several aspects have been explored, among them risk assessment related to changes [7], automation [8], alignment to business purposes [9], automated generation of IT change plans [10]. Following in this section we discuss some of the most important researches that have dealt with failures in changes.

Machado *et al.* [3] have proposed a solution to generate *rollback plans* prior to the deployment of changes. This solution generates a *rollback plan* for each atomic activity (reversible) of the change, *i.e.*, it generates a plan to undo the changes made during the execution of a CP. When the *Deployment System* (a system that performs all the changes specified in a CP) detects a failure, the *rollback plan* associated with the activity that triggered the failure is then executed. This rollback plan will revert the changes performed and return the managed infrastructure to a consistent state. However, in this approach the cause of the failure is not identified. Moreover, the solution does not reuse the acquired knowledge, allowing that in another execution of activity, the failure can occur again, but without any special treatment.

The discipline of problem management is responsible for managing the lifecycle of failures, also called Problems and/or Incidents [5]. Some concepts, relationships, and flows that help in the diagnosing faults are presented. However, this information is described at a high level of abstraction. Aiming to cover some deficiencies of ITIL, Jäntti and Eerola [11] have proposed a conceptual model for service-oriented problem management with support for problems and defects of software. This model allows the documentation of incidents and/or problems throughout the lifecycle of a software. Furthermore, it allows problems management both in proactive and reactive ways. However, methods for the identification the root cause of failures were not considered, resulting only in the documentation of the diagnosis result.

Currently, many processes, such as problem and incident management, are implemented manually by organizations. Therefore, Gupta *et al.* [12] proposed a solution for automating the flow of the process of incident management, using techniques of information integration and machine learning. Among these techniques, a method deserves special attention because it allows the correlation between incidents and Configuration Items (CI's). This correlation allows the diagnosis of the components that presented failures when incidents are reported. Based on information from a dictionary, keywords are indexed for both incidents and their relationships. By comparing the keywords this solution is able to identify: similar incidents or recurring; possible component that failed; and solutions used in the past. Using information about past experiences this solution allows the identification of the CI that failed. However, to identify the CI that failed might not enough, because it may have failed for several different root causes.

In reviewing other areas related to failures, we can observe great progress. Appleby *et al.* [13] have proposed the Yemanja, a mechanism based on event correlation to diagnose multi-layers faults. Such mechanism performs diagnosis in complex scenarios of fault propagation. Moreover, events from networks of low-level can be easily correlate with warnings of high-level applications. This correlate can detect violations in the quality of services. Steinder and Sethi [14] have proposed a method for fault diagnosis on end-to-end services, based on analysis of symptoms mapped on causality graphs and its dependencies. This method allows diagnosis of problems of both availability and performance in multiple layers of the protocol stack. However, such research is focused in finding the cause of failure in an autonomous fashion. Our research is situated in the area of IT change management, where different sources of failures are considered that cannot be diagnosed automatically, such as human errors.

Many recent investigations have focused in incident and problem management, however none of those allows the identification of root causes of failures that occur in the implementation of changes. Moreover, few researches investigate the possibility of learning from past experiences. To address these shortcomings, in this paper we propose a conceptual solution, extending the CIM model, which allows the identification of the root cause of failures in the execution of Change Plans (CP).

III. Proposed Solution

The solution proposed in this paper for identifying the root cause of problems is based on information from diagnostic of previous failures collected from the IT environment. Based on such information and also considering the current state of the managed infrastructure, the proposed solution is able to interactively help IT operators in finding the root cause of failures.

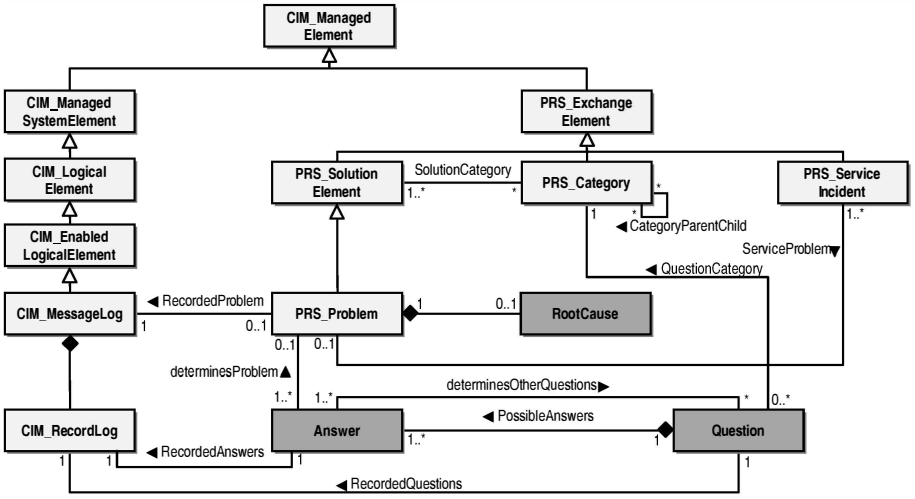


Fig. 1. CIM model extended to represent diagnostic information

In this paper, we assume that failures during the implementation of changes are recurrent, *i.e.*, by observing the history of executions of a CP and associated diagnostic information, it is possible to analyze failures occurred in the past and identify likely Root Causes (RCs) for subsequent failures. We assume also that the remaining stages of the problem management will be performed by a system or manually by operator. In other words, the main focus of this work is the identification of RC and not to find the best solution.

In this section, the solution for interactive identification of RC of problems/incidents in change management is presented. This solution considers two factors that allow the dynamicity of the selected questions: (*i*) the current state of the managed infrastructure, that makes it possible to select of relevant questions related to CIs involved; and (*ii*) the history of previous diagnostics, that allows the identification of questions most frequently used in the past. In this section, in a first moment, the model that allows the representation of diagnostic information necessary is described. Afterwards, it will be presented the conceptual architecture of our solution is presented. Finally, we introduce the the display process of identification along with algorithms that enable interactive diagnosis.

A. Information Model

According to ITIL, organizations should maintain a database that represents the current state of their IT infrastructure. There are some models that include much of the information necessary for the precisely describe humans, software, hardware, and many other Configuration Items (CIs). The Common Information Model (CIM) proposed by the Distributed Management Task Force (DMTF) [6] is widely used for this purpose. This model allows describing in high level of detail the elements of an IT infrastructure, such as, computer systems, humans, or processes, and also the relationships among all of them. A specific set of classes from CIM is designed to represent information about support, including incident and problem reports. Such classes are referenced by the DMTF as an extension, and are called Problem Resolution Standard (PRS). However, this model does not provide classes for the representation of Root Causes. This hinders the reuse of knowledge obtained from previous diagnoses. Therefore, it is necessary to extend the CIM model in order to enable the association of CIs to information about diagnosis and repair.

To implement our proposed solution and provide system support for identification of RC, it is necessary to extend the CIM/PRS model. Our expansion of this model basically consists of adding three new classes: *Question*, *Answer* and *Root Cause.* In Figure 1, a simplification of the proposed model is presented. The extended classes are displayed in dark gray, and the classes of the model CIM/PRS in light gray.

A *RootCause* is aggregated to the class *CIM_PRS_Problem.* This aggregation enables the association of causes with a problem, as well as to several incidents. A problem may or may not have an RC, and consequently an incident does too.

A *Question* is an aggregation of possible *Answers.* A *Question* is also associated with a *PRS_Category*, which allows identifying the level of the *Question*, *i.e.*, if a question is related to a particular device type (*e.g.*, "Network Devices"), or if it is in a more generic scope (*e.g.*, "Devices"). This property allows higher-level questions to be selected first. According to ITIL, a category could have up to three levels of parent/child categories (four categories in total). For example, the CI "Apache" belongs to "Software" and "Web Server" categories. Moreover, the "Web Server" category is dependent on a higher level category, in this case, "Software". Therefore, we considered the Web Server category belonging to Level 2 and the Software category belonging to Level 1.

The *Answer* of a *Question* may or may not enable the selection of another *Questions.* A set of certain *Questions* and *Answers* can determine a *PRS_Problem* and, consequently, it is *RootCause.* Both the questions selected by the *Diagnosis System* and the answer selected by the operator are recorded in a *CIM_RecordLog. CIM_MessageLog* is an aggregation of several *CIM_RecordLogs.* Moreover, a *CIM_MessageLog*

contains all the questions and answers selected for a diagnosis, as well as the RC identified.

B. Conceptual Architecture

Based on a generic architecture of a change management, the *Diagnosis System* (*DS*) was introduced to support the identification of the root cause of failures. In Figure 2, an overview of our conceptual architecture is presented. The main components, actors, and interactions among them are shown in this figure. The components introduced in this work are highlighted with gray background.

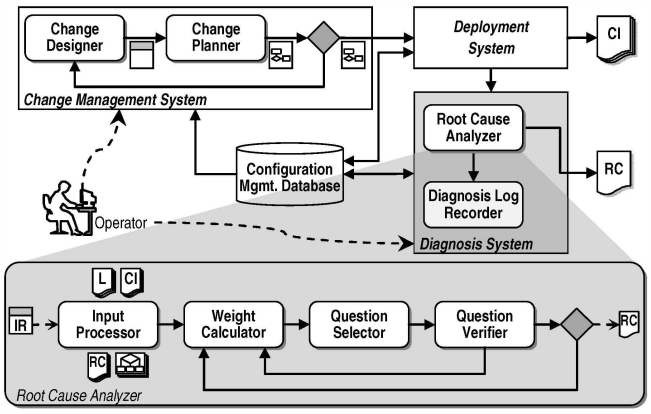


Fig. 2. Conceptual Architecture of the Proposed Solution

An operator interacts with *Change Management System* to create a Request For Change (RFC). The initial specification of an RFC is performed more precisely in the component called *Change Designer*. In general, an RFC describes what changes should be performed in the managed infrastructure, the CIs affected (*e.g.*, firewall, switches, services, and applications) and the business purposes to be achieved. However, an RFC does not specify the details of implementation. These details are specified by the operator in the preliminary Change Plan by interacting with the component called *Change Designer*. This preliminary plan consists of a workflow of activities (or actions) that describe in a high level of abstraction the steps necessary to materialize the change over the infrastructure.

The preliminary Change Plan is refined in the component called *Change Planner*. The result of this refinement is a workflow of low level activities that can be effectively performed over the CIs. Such workflow of activities is called Change Plan (CP) [15]. At this point, other solutions may apply (*e.g.*, rollback support [3], risk management [7] [16], and/or human resources allocation [17]) but will not be detailed in this paper. Afterwards, the CP is analyzed by the operator, which will decide to allow it is execution as is, or to return the CP to the *Change Designer* for possible improvements or adjustments.

Assuming that the operator has approved the CP and submitted it to the *Deployment System*. This system basically implements the changes described in CP over the managed IT infrastructure. At the same time, relevant information about the changes being implemented are stored to log. For every activity of CP that is completed, the *Deployment System* updates information in the Configuration Management DataBase (CMDB) to maintain this database consistent with the current status of CIs.

In a system that supports the identification of RC, the failure is detected by the *Deployment System*, and then the operator may decide to identify the RC or execute another solution. By choosing for the identification of RC, the operator interacts with the *Diagnosis System*, answering questions with the answers available. This selection of questions is performed by the component called *Root Cause Analyzer* that is further detailed in next subsection. The information that allows the identification of RC (*e.g.*, questions, answers, and the RCs) are stored in the CMDB. After determining the RC, the *Root Cause Analyzer* informs to the operator that checks whether the identification was successful or not. In the case of a negative response, the operator can return to the identification process, to perform a new diagnostic.

By completing the process of identifying the RC of the failure, relevant information (*e.g.*, selected questions, answers, and RC identified) about diagnosis made by the operator are stored in the system logs by the component called *Diagnosis Log Recorder*. This information will be reused by the system in future diagnostics and can also be useful for other modules or solutions.

C. Root Cause Analyzer

In order to help operators on identifying the RC of failures in changes, we have adapted an interactive approach. Basically, the interaction happens between the *Diagnosis System* and the operator, as system selects a question to be answered by the operator. Such interaction can dynamically generate a new diagnostic workflow for every execution, based on the answers provided by the operator. This approach allows the system to adapt the selection of questions at runtime: the problem/incident being diagnosed; the affected IT infrastructure; the history of executed diagnoses; and answers previously informed by operators.

Assuming that a failure occurred during the execution of a Change Plan (CP), the operator must create an Incident Report (IR). The IR contains relevant information for the identification process of the RC, for example, the CI impacted by the failure, the responsible for the incident, the priority, and the identification of the IR. Such report can be manually generated by the operator or automatically by the system. In the latter case, the operator is only responsible for identifying the Root Cause (RC) of failures detected by the *Deployment System.*

The *Root Cause Analyzer* is the component of the *Diagnosis System* responsible for identifying the RC of the failure. As shown in Figure 2, for a better understanding, this component is divided into four modules: *Input Processor*, *Weight Calculator*, *Question Selector*, and *Question Verifier*.

1) Input Processor: Based on the Configuration Item (CI) informed in the IR, the *Input Processor* module identifies the dependents CI. This will list all the elements that may have

caused or even influenced in the failure. These dependencies are mapped by the CIM. After identifying all the elements that may have influenced in the failure, this module seeks the categories of each CI. Assuming that the category identified has multiple levels, then all categories are selected.

Each RC is associated with a set that contains answers and questions. The identification of an RC occurs when this set is fully satisfied. Therefore, when selecting an RC, the questions and answers that identify the RC are also selected.

Both CIs and RCs are associated with categories. Therefore, given a category, it is possible to list the entire infrastructure that may have caused the failure. Also is possible to identify all RCs based on the CI informed by the operator on the IR. After having listed all possible RCs for failure occurred, the logs of past diagnoses are identified. When finalizing a diagnosis, the RC identified is recorded in the log. Through this attribute it is possible to list all the previous diagnoses of a specific RC. This allows the history to be consulted and considered in new diagnoses.

2) Weight Calculator: To have a better chance of identifying the correct RC, the RCs that have a greater number of correct diagnoses are prioritized. This information is obtained through the logs of the *Diagnosis System*. The weights assigned to categories, questions, and answers are calculated based on the weight of RCs associated with these. The weight of an RC is calculated by the number of previous diagnoses on which was RC has been correctly identified. Assuming that an RC has been identified as the cause of failure in 11 different diagnostics, then the weight of this RC in new diagnostics is 11.

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,Log)
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**

As presented in Algorithm 1, the weight of the categories, questions, and answers is calculated based on the weight of the RCs associated. For each RC of the set of possible is assigned a weight, based on the logs (Line 2). Each question (Line 4), answers (Line 6), and associated category (Line 10) receives the same weight. But in these cases the weight of the RC is added to the current weight. This addition enables a question or category that is associated with several RCs have a higher weight than one that is associated with only one of these RCs. However, the response will only have added the weight of the RC, if it belongs to the set of questions and answers from this RC.

3) Question Selector: After assigning weights to all questions, answers and categories previously identified. The *Question Selector* module selects a question to be asked to the operator. This selection is performed based on the weights. First, the component selects the category that has the greatest weight. Within this category, the question that has the greatest weight is selected. When more than one question has the same weight, the question with higher level will be selected, *i.e.*, the question that is not dependent of a determined answer or that is associated with higher level category.

4) Question Verifier: Based on the history of executions, it can identify how many times the question was selected and what response was given by operators. This history is mapped in the weights of questions and answers. After being selected, a question may be considered as obvious or not. If considered obvious, the question does not have to be answered by the operator. The answer with the higher weight will be automatically assumed by the *Diagnosis System*. Otherwise it is requested the operator intervention that must select one of the available answers.

Algorithm 2 Verification of Question

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

1: $R \leftarrow null$
2: **if** $Weight\_P >=$ threshold **then**
3: **for all** $A \in$ set of answers from P **do**
4: **if** $Weight\_A >= 80\%$ of $Weight\_P$ **then**
5: $R \leftarrow A$
6: **end if**
7: **end for**
8: **end if**
9: **if** R is $null$ **then**
10: $R \leftarrow$ OPERATOR_INTERVATION(P)
11: **end ifreturn** R

The obviousness of a question can be verified by analyzing the weight of it is answers. The steps to perform this verification are presented in Algorithm 2. The first condition that must be satisfied is the weight being greater than or equal to the threshold (Line 2). In this paper, we consider a threshold of weight 10. It is worth mentioning that we derived this threshold from previous experiments. According to the needs of each organization this value can be adjusted, in order to adapt the solution for different environments. This threshold allows RCs with few executions are not considered obvious. After the *Question Verifier* module checks whether any of the answers have weight greater than or equal to 80% by weight of the question (Lines 3 and 4). This percentage is proposed by Pareto Analysis technique. This technique is presented in

the ITIL and permits separating important and potential causes from more trivial issues [5]. When both conditions are satisfied the obvious answer is assumed by *Question Verifier* (Line 5). However, if a condition is not met, the question is sent to the operator, that must answer using one of the available answers (Lines 9 and 10).

Assuming then that the question "The network card is configured?" has a weight of 23, and the answer "Yes" has weight 19. So this question will not be sent to the operator, because it has more than 10 executions and in 82.60% of cases the answer was the same. The *Question Verifier* takes this answer as if the operator had chosen it.

Based on the response selected, the *Root Cause Analyzer* makes a refinement on the set of possible RCs. The RCs and questions that cannot be identified are ignored. The weights of the questions, answers, and categories are recalculated with the remnants RCs after refinement. Then the other components are executed.

After completing the diagnosis, an RC is identified. This RC is reported to the operator that informs to the *Diagnosis System* if the diagnostic was successfully completed. If the answer is affirmative, the identification process is terminated. The questions, answers selected, identified RC, and the information about the incident are stored in log. Otherwise, the process is restarted, but ignoring the *Question Verifier* module. This is done to avoid being assigned a default answer to a question that in this diagnosis may be different.

IV. Case Study

In order to evaluate the technical feasibility of the proposed solution, different scenarios were analyzed. In this section, the case study presented is focused on a failure occurred in the execution of an RFC instantiated in two distinct IT infrastructures. As a result of these analysis we intend observe the effectiveness (check if the root causes are determined correctly) and the dynamicity of the proposed solution (check if the resulting diagnostic workflows are generated based on (*i*) the infrastructure where failure occurs and (*ii*) according to prior documented answers).

In the Figure 3, the main activities – in high level of abstraction – involved in the RFC are presented. This RFC has the purpose to install a webmail service (provided by the software Squirrelmail) and its dependencies. For example, before installing Squirrelmail it is necessary to have installed a web page service (Apache) and an e-mail service (Exim). In the first scenario, the activities of the RFC are installed on a single computer (shown in Figure 4(a)). On the other hand, in the second scenario, the activities related to the e-mail service (Exim) are installed on a different server where the web page service is hosted (shown in Figure 4(b)). For both scenarios, we considered that the activity *Verify Webmail Access* failed.

According to the Diagnosis System presented in Figure 2, an Incident Report (IR) is created when a failure in the activity *Verify Webmail Access* is detected. Further information about IR is provided in Subsection III-C. In the next step, the

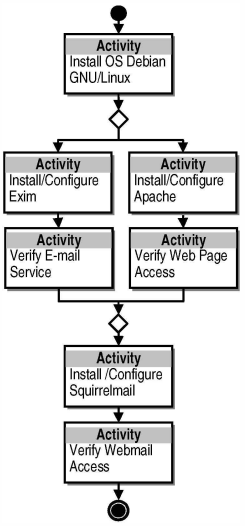


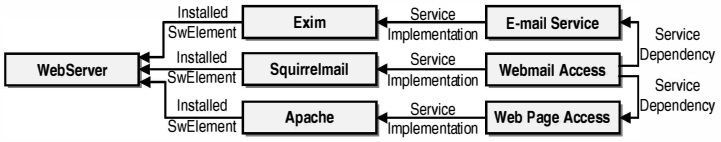
Fig. 3. Change Plan for the installation of a webmail service

Input Processor module identifies the dependent CIs related to the activity that had failed (*Verify Webmail Access*). In Figure 4(a) and 4(b), some dependencies among the CIs can be observed. *Service Dependency* represents that a service depends on another service to work properly; *Service Implementation* represents a service that depends on a software; *InstalledSwElement* represents dependency of a software element on the computer where it is installed; and *Hosted Service* represents the dependency of a service on a hardware. These and other dependencies are modeled using the CIM model.

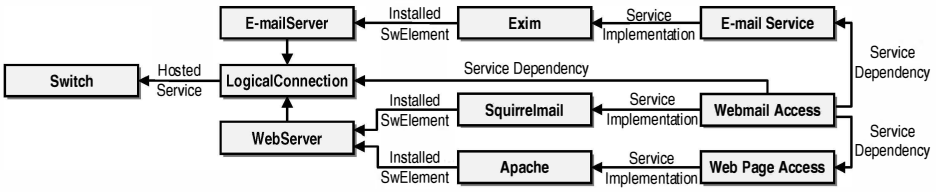
As presented in Subsection III-C1, after the dependencies are identified, the *Input Processor* selects the categories of each CI. Based on these categories the potential Root Causes (RCs), questions, answers, and logs of previous diagnoses are identified. Afterwards, based on the weight of the RCs (frequency that a RC is diagnosed) the weights for the categories, questions, and answers are calculated by the *Weight Calculator* module. In Table I, weights computed for CIs and theirs respective categories for scenario 1 are presented. In addition, in Table II, the same information for scenario 2 is presented.

As mentioned before in Subsection III-C3, the *Question Selector* module selects questions to be answered by the operator respecting to the weights of each category. In the first scenario, the order of level 1 categories according to their weights is *Software*, *Service*, and *System*. Whereas in the second scenario, the order of level 1 categories is *Software*, *Devices*, *Network*, *System*, and *Service*. It is also important to notice that weights are recalculated every interaction and also that the second scenario, by having a larger number of CIs, will have listed a greater number of categories, questions, answers and RCs.

For example, the CI *E-mail Service* belongs to a category of level 2, where the highest-level category is called *Service* with weight 25, and the lowest-level category is *Email* with weight 17. It is important to mention that when a category is



(a) Infrastructure with a single server



(b) Infrastructure with two servers

Fig. 4. Infrastructure instantiated for the evaluation

TABLE I
CIs, CATEGORIES AND ASSOCIATED WEIGHTS FOR SCENARIO 1

CI	Category	Weight
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	52⇒25
Squirrelmail	Software ⇒ Webmail	52⇒9
Apache	Software ⇒ Web Server	52⇒18
WebServer	System ⇒ Computer System ⇒ Web Server	21⇒16⇒7

TABLE II
CIs, CATEGORIES AND ASSOCIATED WEIGHTS FOR SCENARIO 2

CI	Category	Weight
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	52⇒25
Squirrelmail	Software ⇒ Webmail	52⇒9
Apache	Software ⇒ Web Server	52⇒18
WebServer	System ⇒ Computer System ⇒ Web Server	26⇒21⇒5
E-mail Server	System ⇒ Computer System ⇒ Mail Server	26⇒21⇒7
Logical Connection	Network	41
Switch	Devices ⇒ Network Devices	48⇒36

analyzed, its weight is a result of a computation of its RCs weights adding it is lower categories weights, as previously presented in the Algorithm 1.

In the next step, the *Question Selector* module selects the question based on the previously defined criteria (weights), and the *Question Verifier* checks the obviousness of the selected question. Based on the operator's answer, the set of possible RCs is depurated (RCs and questions are discarded since they cannot be identified anymore). This process repeats until a RC is identified, then it is informed for the operator that determines the success of the diagnosis.

The diagnostic workflows resulting for scenarios 1 and 2 are shown in Figures 5(a) and 5(b), respectively. Questions that appear with gray background were answered automatically by the system. It occurs because the *Question Verifier* considered these answers obvious, *i.e.*, the operator intervention was not requested. Information about the weight (number of executions for each question) and the percentage of the same answer are presented beside the questions. Moreover, the line *Operator's Answer* represents the answer of the operator.

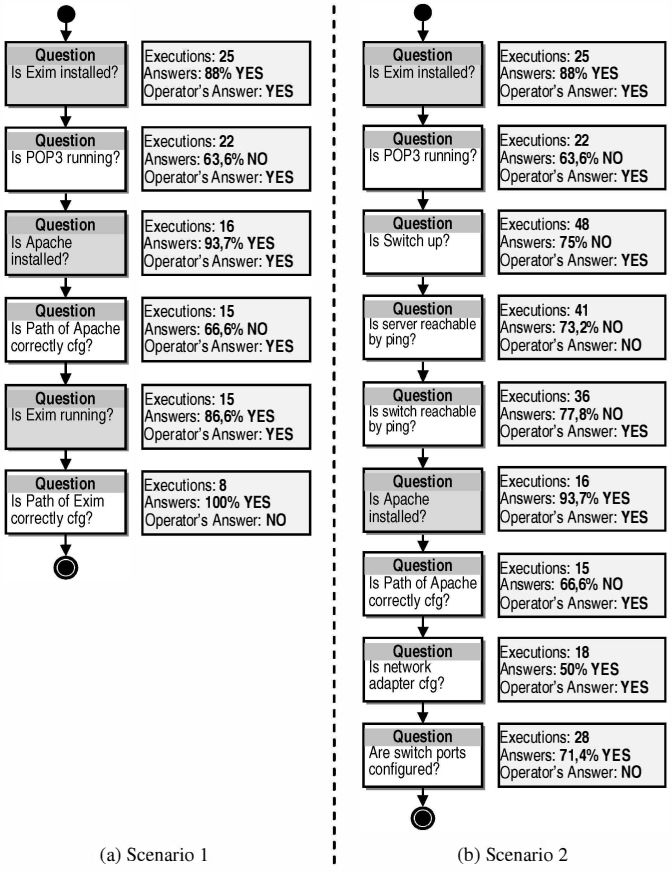
Each answer informed by the operator has direct influence on the selection of the next question. It results in a sequence of dynamic and interactive diagnostic, since the complete diagnostic workflow is generated only after the end of the diagnostic process.

Once the operator finishes answering the questions in both scenarios, the *Diagnosis System* identifies correctly the RCs. For scenario 1 RC *Path to e-mail files is wrong* was identified and for scenario 2 the system pointed to the RC *The ports of the Switch were not configured properly.*

It is important to emphasize that these two different RCs – found in different scenarios – were identified from failures in the same activity. Moreover, it is possible to notice that workflows generated for each scenario were also different. In addition, we would like to highlight that our solution would not be able to identify a RC when it is not present in CMDB, *i.e.*, never documented RCs.

V. CONCLUSIONS AND FUTURE WORKS

The identification of Root Causes (RCs) process represents a critical step in the IT infrastructures operation and manage-



(a) Scenario 1 (b) Scenario 2

Fig. 5. Diagnostic workflows generated by the solution

ment. However, the existing solutions to support this process do not use methods that can easily reuse the knowledge acquired. Consequently, failures in the deployment of changes may occur without operators having a clear understanding about what is causing these problems. To address this issue, in this paper we proposed a solution to help in the identification of RCs of failures in IT changes using dynamically questions and answers. This approach makes the diagnostic process more interactive and dynamic.

The results obtained during the case study conducted have shown the benefits of using our proposed solution to identify the RCs of activities of changes that had failed. As previously presented, the solution considers the nuances of the environment and adapts the process to variations in the infrastructure. From the same failure, the solution generates different diagnostic workflows, since it considers the CIs involved, historic of executions and the answers of the operator. Finally, the solution proved to be feasible and effective.

As future work, we envision four main improvements to our current solution: (*i*) investigate strategies that consider other metrics to improve the *Question Selector* module, *e.g*, considering the age of past diagnoses upon selection in order to prioritize recently identified RCs; (*ii*) adopt heuristics that allow to optimize workflows generated to identify the root causes; (*iii*) investigate the use of dependencies to diagnose dependent CIs aiming to improve the bootstrapping of the Diagnosis System; and (*iv*) extend the process to identify root causes for other scopes, *e.g.*, applying the same methods to manage incidents in service operations/support.

REFERENCES

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

[2] ITIL, "ITIL - Information Technology Infrastructure Library. Office of Government Commerce (OGC)," 2009, Available: http://www.itil-officialsite.com/. Accessed: aug. 2010.

[3] G. Machado, F. Daitx, W. Cordeiro *et al.*, "Enabling rollback support in IT change management systems," in *Network Operations and Management Symposium, 2008. NOMS 2008. IEEE*, April 2008, pp. 347–354.

[4] W. Cordeiro, G. Machado, F. Andreis *et al.*, "ChangeLedge: Change design and planning in networked systems based on reuse of knowledge and automation," *Computer Networks*, vol. 53, no. 16, pp. 2782 – 2799, 2009.

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

[6] DMTF, "Distributed Management Task Force: Common Information Model. Distributed Management Task Force (DMTF)," 2009, Available: http://www.dmtf.org/standards/cim. Accessed: aug. 2010.

[7] J. Sauve, R. Santos, R. Reboucas, A. Moura, and C. Bartolini, "Change priority determination in it service management based on risk exposure," *Network and Service Management, IEEE Transactions on*, vol. 5, no. 3, pp. 178 –187, september 2008.

[8] A. Brown and A. Keller, "A best practice approach for automating it management processes," in *Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP*, 3-7 2006, pp. 33 – 44.

[9] A. Moura, J. Sauve, and C. Bartolini, "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*, vol. 46, no. 10, pp. 148 –153, october 2008.

[10] A. Keller, J. Hellerstein, J. Wolf, K.-L. Wu, and V. Krishnan, "The champs system: change management with planning and scheduling," in *Network Operations and Management Symposium, 2004. NOMS 2004. IEEE/IFIP*, vol. 1, 23-23 2004, pp. 395 –408 Vol.1.

[11] M. Jantti and A. Eerola, "A Conceptual Model of IT Service Problem Management," in *Service Systems and Service Management, 2006 International Conference on*, vol. 1, Oct. 2006, pp. 798–803.

[12] R. Gupta, K. Prasad, and M. Mohania, "Automating itsm incident management process," in *Autonomic Computing, 2008. ICAC '08. International Conference on*, 2-6 2008, pp. 141 –150.

[13] K. Appleby, G. Goldszmidt, and M. Steinder, "Yemanja-a layered event correlation engine for multi-domain server farms," in *Integrated Network Management Proceedings, 2001 IEEE/IFIP International Symposium on*, 2001.

[14] M. Steinder and A. S. Sethi, "Probabilistic fault diagnosis in communication systems through incremental hypothesis updating," *Computer Networks*, vol. 45, no. 4, pp. 537 – 562, 2004. [Online]. Available: http://www.sciencedirect.com/science/article/B6VRG-4C3S1PT-2/2/bd09665fe2632ebb8b0255832e9c5c4f

[15] W. L. C. Cordeiro, G. Machado, D. F.F. *et al.*, "A template-based solution to support knowledge reuse in IT change design," in *Network Operations and Management Symposium, 2008. NOMS 2008. IEEE*, April 2008, pp. 355–362.

[16] J. A. Wickboldt, L. A. Bianchin, R. C. Lunardi *et al.*, "Improving it change management processes with automated risk assessment," in *XII IFIP/IEEE International Workshop on Distributed Systems: Operations and Management (DSOM 2009)*, 2009.

[17] R. C. Lunardi, F. G. Andreis, W. L. d. C. Cordeiro, J. A. Wickboldt, B. L. Dalmazo, R. L. d. Santos, L. A. Bianchin, L. P. Gaspary, L. Z. Granville, and C. Bartolini, "On strategies for planning the assignment of human resources to it change activities," in *Network Operations and Management Symposium, 2010. NOMS 2010. IEEE*, apr. 2010, pp. 248 –255.

ANEXO B ARTIGO PUBLICADO - SBRC 2011

Neste anexo, o artigo intitulado “Identificação Interativa da Causa Raiz de Problemas em Execuções de Mudanças de TI” é apresentado. Esta foi a segunda publicação sobre o tema de pesquisa, apresentado nesta dissertação, em eventos científicos renomados. Além do melhor detalhamento da solução para identificação da causa raiz, neste artigo também são introduzidas melhorias que permitem otimizar a seleção das perguntas utilizadas. É importante salientar que este artigo introduz o conceito de diagnósticos frustrados e, consequentemente, inclui penalizações no cálculo dos pesos dos elementos. Por fim, um estudo de caso é conduzido a fim de comparar os resultados obtidos através da utilização das duas versões da solução proposta.

- **Título:**
  Identificação Interativa da Causa Raiz de Problemas em Execuções de Mudanças de TI
- **Conferência:**
  XXIX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2011)
- **URL:**
  `http://sbrc2011.facom.ufms.br/`
- **Data:**
  30 de Maio à 03 de Junho de 2011
- **Local:**
  Centro de Convenções Rubens Gil de Camillo, Campo Grande, Mato Grosso do Sul, Brasil
- **Biblioteca Digital da SBC:**
  `http://www.lbd.dcc.ufmg.br/colecoes/sbrc/2011/0062.pdf`

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

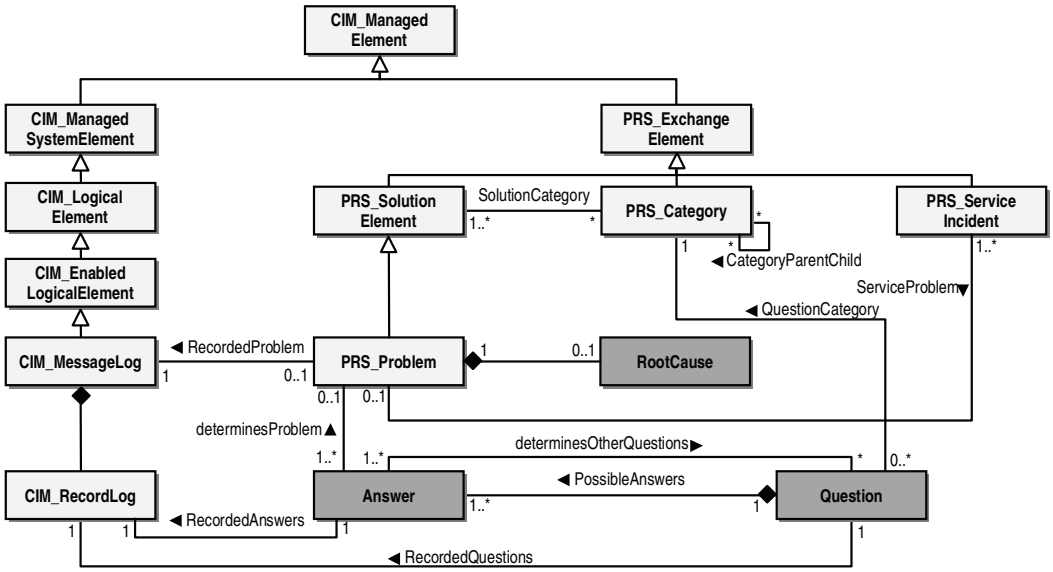


Figura 1. Extensão do modelo CIM para representar informações de diagnóstico

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.

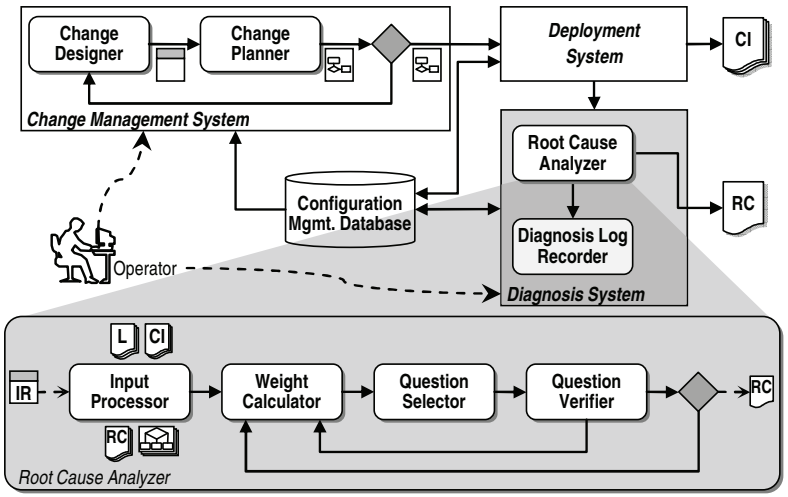


Figura 2. Arquitetura Conceitual da Solução Proposta

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 ∈ set of root causes from S do
2:      Weight_RC ←WEIGHT_CALCULATE(RC,Logs)
3:      for all Q ∈ set of question from RC do
4:          Weight_Q ←WEIGHT_ADD(Q,Weight_RC)
5:          for all A associated with RC ∈ set of answers from Q do
6:              Weight_A ←WEIGHT_ADD(A,Weight_RC)
7:          end for
8:      end for
9:      for all C ∈ set of categories from RC do
10:         Weight_C ←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).

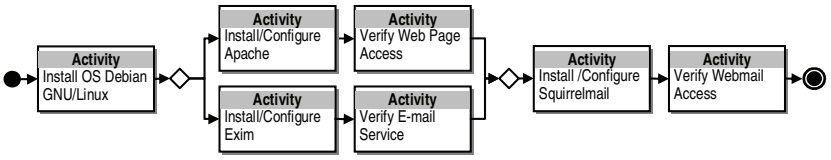
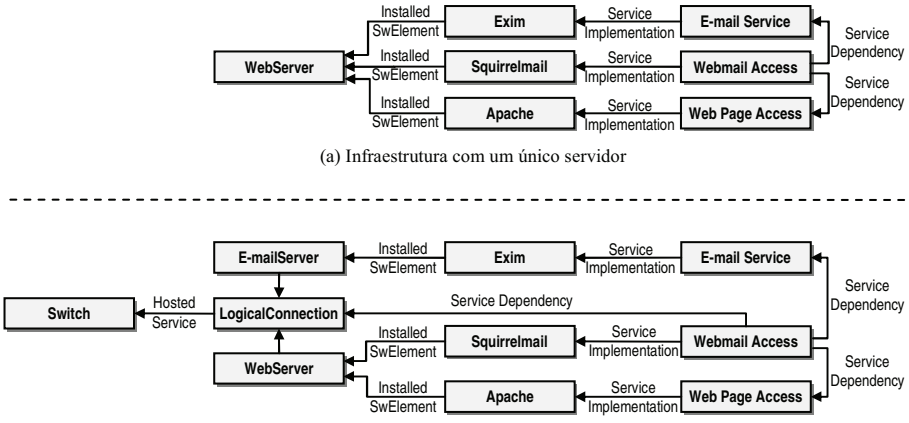


Figura 3. *Change Plan* para a instalação de um serviço de *webmail*

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.



(a) Infraestrutura com um único servidor

(b) Infraestrutura com dois servidores

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*.

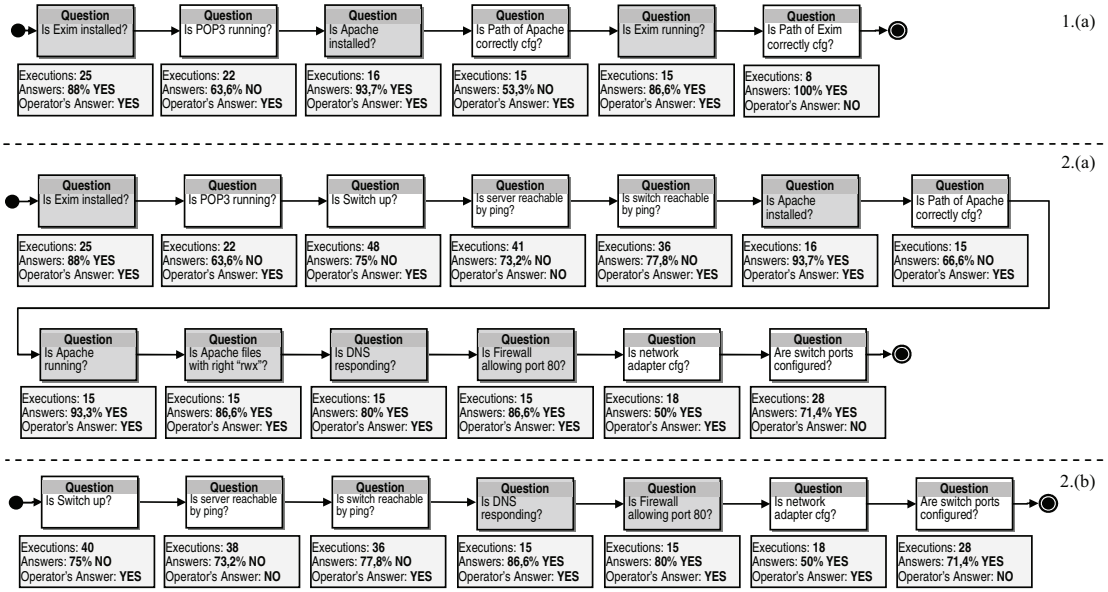


Figura 5. Workflow de Diagnósticos gerados pela solução

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.