Identifying the Root Cause of Failures in IT Changes: Novel Strategies and Trade-offs

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

Roben C. Lunardi
Federal Institute of Rio Grande do Sul, Brazil
roben.lunardi@restinga.ifrs.edu.br

***Abstract*—Despite the Change and Problem Management have received significant attention from the academic community in recent years, the developed solutions do not identify the root cause of failures in IT Changes and, in some cases, only detect software failures. To address this, in this paper, we introduce four strategies to identify root cause of problems based on an interactive approach, in which the Diagnosis System questions a human operator. The strategies introduced and evaluated in this paper are built upon a system we have developed previously, but whose root cause identification was more rudimentary. A case study that uses the improved solution is conducted for the purpose of analyzing the diagnostics generated. Thus, it was possible to compare the diagnostics generated by each strategy, identifying any trends.**

I. Introduction

The dynamic and highly competitive environment in which modern organizations are inserted has significantly increased the importance of Information Technology (IT). The increasing complexity of IT infrastructures (*e.g.*, network and services) turned Information Technology Service Management (ITSM) essential to any organization [1]. In this context, guidelines of best practices and processes have been developed, seeking better performance out of provided products and services. Among the most widely accepted proposals, it is worth mentioning the Information Technology Infrastructure Library (ITIL) [2].

In ITIL, Change Management and Problem Management are two addressed processes quite important organizations. Change Management defines how changes should be planned, scheduled, implemented, and evaluated in IT environments [3]. However, despite the significant improvements that the adoption of Change Management can provide, the occurrence of failures during a change execution cannot be neglected. To address possible failures, Problem Management aims to: (*i*) prevent problems and resulting incidents from happening, (*ii*) eliminate recurring problems, and (*iii*) minimize the impact of incidents that cannot be prevented. In order to achieve these goals, the reuse of operator's knowledge, on IT processes, has became fundamental because it allows simplifying the procedures of detecting the route cause of failures and, hence, minimizes financial losses and maintenance costs [4].

Both Change Management and Problem Management have received significant attention from academia in recent years. Aspects such as changes and failures have been exploited by several researches, including, for example, automating problem classification [5], rollback plans [6], risk analysis [1], and identification of software failures [7]. The aforementioned researches, however, do not identify the root cause of problems and, in some cases, are specific solutions limited to the detection of software failure, ignoring hardware or human ones. Furthermore, these solutions describe actions in a quite static way, this description limits the identification of similar problems and the adaptation of cases to current infrastructure.

In this paper, we introduce four strategies to identify root cause of problems based on an interactive approach, in which the Diagnosis System questions a human operator. The four strategies introduced and evaluated in this paper are built upon a system we have developed previously [8] but whose root cause identification was more rudimentary, based solely on logs of past diagnoses. The strategies employed new determine which question has the larger probability of identifying the root cause. Each strategy exploits different ways to select questions to diagnosis process, such as, frequent questions, popular questions, and questions with the larger recent history. A case study that uses the improved solution is conducted for the purpose of analyzing the diagnostics generated. It is important to notice that each strategy is applied in the same scenario. Thus, it was possible to compare the diagnostics generated by each strategy, identifying any trends.

The rest of the paper is organized as follows. In Section II we discuss key research efforts related to failures and Change Management. In Section III we detail the improved solution and the major components that interact with the proposed strategies. In Section IV we introduce the novel strategies for selecting questions in each interaction. Finally, a case study conducted to evaluate and compare the proposed strategies is described in Section V, whereas Section VI concludes the paper with final remarks and perspectives for future work.

II. Related Work

The ITSM area has received great attention from the scientific community in recent years. Several aspects have been investigated, for instance, the automation of change plans [9] and ITSM processes [10], assessment of risks associated with changes [1], the detection of conflicting change plans [11], reuse of knowledge [12], and alignment to business purpose [13]. Following, we discuss some of the most important studies in the ITSM area particularly related to failures in changes.

In seeking to automate the problem categorization, Diao *et al.* [5] proposed a collaborative solution based on rules. In this

978-3-901882-50-0

solution two types of users are defined: authors, experienced professionals who create the rules used in the system, and consumers, who use these rules to classify the reported problems. In order to simplify and make the classification process more agile, the rules are divided into inclusive and exclusive. First, for each reported problem exclusive rules are executed. Such rules, when satisfied, exclude the categories of the set of valid codes. Soon after, the inclusive rules are executed. In this case, when the conditions are satisfied, the reported problem is classified as a linked category.

In another research, Bartolini and Stefanelli [14] address the alignment of IT processes with business objectives, defined in *Service Levels Objectives*. This research describes a decision support system applicable in various fields, such as change and incident management. This system consists of components that can be reused, besides guidelines and standards that enable the development of specific solutions. The application of these solutions can reduce costs and speed up the execution process. Furthermore, an algorithm is described for prioritizing incidents, which uses business goals to set the priority of resolution of reported incidents [15].

In a recent study, Agarwal and Madduri [7] developed a system for automating the root cause identification in asynchronous failures, generated during the execution of changes. This solution, however, detects only breaches in Service Level Agreements (SLAs) or software failures, like the reduction in the agreed quality of services. In addition, the solution detects the SLA breach, even when the breach does not occur at the same time of the change execution. The system uses associations between failures and changes to identify the activity that failed. However, the activity may fail for several reasons, resulting in a group of root causes. Moreover, the solution identifies only software failures and does not consider activities involving other resources, such as, humans.

Despite the attention received by ITSM area from the scientific community in recent investigations, none of the studies allows identifying the the root cause of failures occurred in IT changes. Moreover, few researches explore the possibility of learning from past experiences. To fill this gap, in the following sections, we propose and evaluate four novel strategies for identifying the root cause of failures in IT changes.

III. Diagnosis System

The novel strategies for selecting questions are supported by a conceptual solution, previously proposed [8], based on ITIL processes. This solution enables identifying the Root Cause (RC) of failures occurred during the execution of IT changes. To facilitate its adoption, the solution is uses the Common Information Model (CIM) [16], a widely known standard that allows to represent widespread IT infrastructures.

Figure 1 depicts the conceptual architecture. The change process begins when an operator interacts with a generic system for the change management, also called *Change Management System*. This interaction aims at generating a Request for Change (RFC) in the *Change Designer*. Soon after that, the RFC is refined by *Change Refiner*. The result of this refinement is a Change Plan (CP). A CP is one *workflow* with low-level activities that may be effectively executed on the IT infrastructure. These and others components are detailed by Lunardi *et al.* [17] and Cordeiro *et al.* [18].

After, the CP is evaluated by a change authority, hidden to improve readability. If disapproved, the CP returns to the component *Change Designer* for possible changes or adjustments. If approved, the CP is then forwarded to the *Deployment System*. In this system, the CP is then deploy on the IT infrastructure in order to reflect the solicited changes. Moreover, the *Deployment System* stores information about the CP deployment in *logs* and maintains an updated view of IT infrastructure in the *Configuration Management Database*.

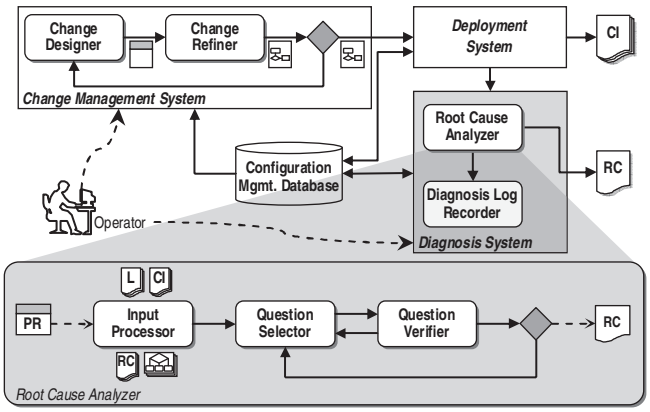

Fig. 1. Overview of the conceptual architecture of Diagnosis System

When the *Deployment System* detects a failure, a Problem Record (PR) is generated and the diagnosis process is started. Thereupon, the operator interacts with the *Diagnosis System* in order to identify the RC of failure reported in PR. The interaction is based on selecting an answer to questions chosen by the system. Based on the operator's answer, the *Root Cause Analyzer* does a refinement in the set of possible RCs. Thus, RCs that can not be identified are ignored, as well as associated informations. This interactive procedure is repeated until identifying a RC. It is important to notice that an expert must create this questions and answers, in order to map his knowledge about diagnoses to the system. To better understand, the *Root Cause Analyzer* component is detailed in the next sub-section.

A. Root Cause Analyzer

The *Root Cause Analyzer* is the component of the *Diagnosis System* responsible for identifying the RC and, therefore, by selecting questions. In order to select a question, the component considers the informations from five sources: (*i*) the Problem Record (PR), (*ii*) the Configuration Items (CIs) belonging to IT infrastructure, which may have influenced or caused the failure, (*iii*) the previous diagnoses where the selected RCs were identified, (*iv*) the answers provided by the operator, and, (*v*) the strategy used for selecting the questions. As can be seen in Figure 1, the component is divided into three modules: *Input Processor*, *Question Selector*, and *Question Verifier*. These modules are briefly described below.

1) Input Processor: This module identifies all elements of infrastructure that may have caused the failure. First, the dependencies of the CI, informed in PR, are mapped. Next, each element of the infrastructure that has a dependency relationship with this CI is stored in a list. Then, this list will contain all dependent CIs. It is important to note that all dependencies among infrastructure elements are mapped by CIM. Finally, the module selects categories in which each CI is associated. Assuming that the identified category has multiple levels, and then categories of all levels are selected.

CIs, RCs, and questions are associated with categories [8]. This association allows listing all possible RCs for an occurred failure, having only of the identified categories in the previous step. Moreover, when a diagnosis is completed, the identified RC is stored in logs. That allows checking previous diagnoses in which a particular RC was identified. Thus, after having listed all possible RCs, logs of previous diagnoses are selected.

Each RC is associated with a single diagnostic set, which contains specific questions and answers. To get a diagnosis done correctly, a diagnostic set, associated with a RC, must be fully satisfied or run until only one RC can be identified. That way, questions and answers that identify each of the RCs, identified in the previous step, also are selected. Both the informations selected by the *Input Processor* and the informations contained in PR are consumed by the *Question Selector*, described in the next sub-section.

2) Question Selector: In order to perform interactive diagnosis, the system selects a question to be answered by the operator. This process is based on information obtained by the *Input Processor*. Aiming to improve the identification of RCs, the more likely questions to identify the correct RC should be prioritized. For this, four novel strategies, which prioritize different questions, are proposed. Among the prioritized questions we consider frequent questions, popular questions, and questions with the larger recent history. The proposed strategies are detailed in Section IV.

Regardless of the strategy selected, the *Question Selector* will select a single question, according to the strategy. Additionally, this module is responsible for calculating the weights to be used by the *Question Verifier*. By having the *logs* of previous diagnostics is possible to calculate the weight of a question, as well its answers weights. The question weight represents how many times it was selected by the system. Moreover, the weight of an answer represents how many times an operator has selected this alternative when queried.

3) Question Verifier: This module determines whether the selected question is obvious or not. A question may be considered obvious through the analysis of its weight and the its answers' weights. If the question is obvious, the module automatically assigns the answer that has the greatest weight to the question. Thus, there is no need for an operator answer. Soon after, the system returns to the selection of a new question by the *Question Selector*. But, if the question is not obvious, the operator intervention is needed, and a answer must be selected among the available alternatives.

To consider a question as obvious, firstly its weight has to be larger or equal to a threshold. In this paper, the threshold considered is equal to 10. This value was derived from experiments performed previously in a simulated environment. According to each organization's need, this value can be adjusted to adapt the solution to different IT environments. The value assigned to the threshold is of fundamental importance, because it allows questions associated with RCs with few executions to be not considered as obvious.

Afterwards, the weight of each of the alternatives is observed. If an alternative has a weight greater or equal than 80% of the question's weight, then the question is considered obvious. In this case, the alternative that satisfies these conditions is automatically considered as an answer for the question analyzed by the *Question Verifier*. However, if no condition is satisfied by any of the alternatives, then the operator intervention is required, which should answer the question by choosing pre-defined alternatives. It is worth noting that the percentage used in this module is proposed by the technique called *Pareto Analysis*. This technique is presented in ITIL and aims to differ potential from trivial causes [4].

At the end of the diagnosis, the identified RC is informed to the operator that determines whether the diagnosis was correctly completed. If correct, the diagnostic process is finished and questions, selected answers, PR, and identified RC are stored in logs by the *Diagnosis Log Recorder*. If the operator informs that the diagnosis was inconclusive or that the identified RC is incorrect, the diagnosis process is restarted, disabling the *Question Verifier*. The system then prevents the removal of questions previously considered obvious, allowing them to be chosen. Persisting the incorrect diagnosis, a warning is generated informing that an expert needs to review the diagnosis. The expert then inserts new RCs and its diagnostic set into the Configuration Management Database and/or makes changes to existing information to eliminate conflicts.

IV. Strategies for Selecting Questions

The improved solution, detailed in the previous section, is fully modular. Thus, the Question Selector module selects only one question, which must be answered by the operator, regardless of the strategy used. Likewise, developed strategies are based on the same inputs: CIs that represent the affected infrastructure, logs of previous diagnoses, categories associated with the CIs, RCs associated with the categories, and the diagnostic set associated with each RC.

For a better understanding of the developed strategies, Table I shows the data processed by the *Input Processor*, for a hypothetical scenario. Each row of the table contains information of a selected RC. For instance, the RC_1 is associated with category C_1 and the identified diagnostic set includes three questions (P_1, P_2, and P_8) and three respective answers (R_1, R_3, and R_{15}). Moreover, RC_1 has: (*i*) 25 completed diagnostics, with 1 in last year and 24 in over a year, and, (*ii*) 12 frustrated diagnostics, with 4 in last year and 8 in over a year.

The improved solution, details in Section III, forms the basis for the identification the root cause of failures in IT changes. Over this solution, it is possible to implement different strategies for selecting questions in the Question Sector module. Such strategies explore different ways to select questions to

TABLE I
SUMMARIZATION OF THE INFORMATION USED BY THE STRATEGIES

RCs	Questions	Answers	Categories	Completed		Frustrated	
				< 1 year	> 1 year	< 1 year	> 1 year
RC_1	P_1, P_2, P_8	R_1, R_3, R_{15}	C_1	1	24	4	8
RC_2	P_4, P_7	R_8, R_{13}	C_1, C_2	4	15	1	2
RC_3	P_5, P_6	R_{10}, R_{12}	C_1, C_3	5	4	2	1
RC_4	P_3, P_9	R_5, R_{17}	C_1, C_2	12	0	2	0
RC_5	P_5, P_{10}	R_{11}, R_{21}	C_1, C_4	3	2	1	0

diagnosis process, such as frequent questions and popular questions. In this section, four novel strategies are detailed. Moreover, the example described above allows demonstrating the selection of questions for each developed strategy.

A. Strategy 1 - Only Completed Diagnostics

The first developed strategy is a simple solution for selecting questions. In summary, weights are assigned to questions, answers, and categories, based on the weights of associated RCs. The RC weight is calculated by the sum of completed diagnostics in which RC was correctly identified. If there are categories, questions, or answer associated with more than one RC, the final weight of these elements is computed by adding the weight of all associated RCs. Thus, an element associated with various RCs has a larger final weight.

Based on the elements' weights, the module selects one question that will be answered by the operator. First, the module selects the category that has a greatest weight. Within this category, the question that has the largest weight is selected. If there are two or more questions with the same weight, the module selects the more abstract question, *i.e.*, the question that is not dependent of a determined answer or that is associated with larger level category. After calculating the weights according to this strategy, categories are ranked as follows: (*i*) C_1, because its weight is 70, (*ii*) C_2 with weight 31, (*iii*) C_3 with weight 9, and, finally, (*iv*) C_4 with weight 5.

Similarly, questions are also ranked as follows: (*i*) P_1, P_2, and P_8 have weight 25, (*ii*) P_4 and P_7 have weight 19, (*iii*) P_5 has weight 14, (*iv*) P_3 and P_9 have weight 12, (*v*) P_6 has weight 9, and, finally, (*vi*) P_{10} has weight 5. In this case, questions associated with C_1 must be selected, because this category has the largest weight. Among questions, P_1, P_2, and P_8 have preference for selection. However, P_2 and P_8 are dependent on P_1. Therefore, the module selects question P_1 to be sent to the operator.

B. Strategy 2 - All Diagnostics

This strategy has been proposed in order to reduce the number of interactions and reuse the knowledge mapped in all diagnoses, whether completed or frustrated. Thus, the calculus of weights is based on all diagnoses stored in the system logs. In this sense, RCs that have frustrated diagnostics are penalized. A diagnostic is frustrated when the system uses at least one question associated with an RC, but at the end of the diagnosis another RC is identified as the real cause of the reported problem. Therefore, the histories of several RCs receive a frustrated diagnostic, but the completed diagnostic will be assigned only to the history of the identified RC. In that way, the RC weight is calculated by the sum of the completed diagnostics subtracted by the sum of frustrated diagnostics. As a consequence, the weights of the other elements are reduced, because they compute the weight of the associated RCs.

By using the Strategy 2 and the data presented in Table I, the weights are assigned to categories that are ranked as follows: (*i*) C_1 has weight 49, (*ii*) C_2 has weight 26, (*iii*) C_3 has weight 6, and, finally, (*iv*) C_4 has weight 4. As can be observed, the order of the categories remains the same. It is important to note that the order could be different, resulting in the selection of questions associated with other categories. Likewise, the questions are ranked by weight as follows: (*i*) P_4 and P_7 have weight 16, (*ii*) P_1, P_2, and P_8 have weight 13, (*iii*) P_5, P_3, and P_9 have weight 10, (*iv*) P_6 has weight 6, and, finally, (*v*) P_{10} has weight 4.

Considering the weights assigned to the categories and questions, questions associated with the category C_1 should be selected. That is so because such category has the largest weight among the selected categories. Among questions, only P_4 is associated with C_1, once P_7 is associated with C_2. In this case, P_4 will be chosen for the operator to answer it. In a simple comparison with Strategy 1, it can be seen that the weights calculated for questions and categories, as well as the selected question by the Strategy 2 are different. Thus, this strategy allows the improved solution adapting itself to the history of diagnoses, because it considers the both completed and frustrated diagnostics.

C. Strategy 3 - Age of Diagnostics

In this strategy, in addition to the penalty for frustrated diagnostics, the elements weights are penalized by the age of diagnostics. The older diagnostics have a smaller value in the weights calculation. This weight penalty is applied to both frustrated and completed diagnostics. The heuristics defined above are formalized in Equation 1. Such equation allows calculating the weight of x element, in which: (*i*) the age of diagnosis is represented by i, (*ii*) the percentage of weight to be used is represented by β_i, and is obtained by subtraction between 100% and the penalty percentage for age i, (*iii*) the amount of completed diagnostics with a determined age is represented by α_i, and, finally, (*iv*) the amount of frustrated diagnostics with a determined age is represented by ω_i.

$$elementWeight_{(x)} = \sum_{i=1}^{10} \beta_i \times (\alpha_i - \omega_i) \quad (1)$$

It is important to notice that if there exist diagnostics with different ages, each age group will receive a percentage of the appropriate penalty, as shown in Table II. For instance, the RC *network card defective* has logs as follows: (*i*) 25 completed diagnostics, which 10 in 1st age, 5 in 4th age, 5 in 7th age, and 5 in 10th age, and, (*ii*) 15 frustrated diagnostics, with 4 in 1st age, 6 in 3rd age, 2 in 8th age, and 3 in 10th age. Applying Equation 1, the following calculation is obtained $elementWeight_{(RC)} = 100\%(10-4) + 90\%(0-0) + 80\%(0-6) + 70\%(5-0) + 60\%(0-0) + 50\%(0-0) + 40\%(5-0) + 30\%(0-2) + 20\%(0-0) + 10\%(5-3)$. After performing the

necessary operations, the final weight of the RC *network card defective* receives the value of 6.3.

Using the detailed information in Tables I and II, it is possible to calculate the elements' weights penalizing the age of diagnostics. Thus, categories are ranked as follows: (*i*) C_1 has weight 18.4, (*ii*) C_2 has weight 14.3, (*iii*) C_3 has weight 3.3, and, finally, (*iv*) C_4 has weight 2.2. Moreover, questions are also ranked as follows: (*i*) P_3 and P_9 have weight 10, (*ii*) P_5 has weight 5.5, (*iii*) P_4 and P_7 have weight 4.3, (*iv*) P_6 has weight 3.3, (*v*) P_{10} has weight 2.22, and, finally, (*vi*) P_1, P_2, and P_8 have weight -1.4. Based on the calculated weights, the module selects the category C_1. Among questions (P_3 and P_9), only P_3 is associated with category C_1. Thus, this question is sent to the human operator.

TABLE II
PENALTY USED TO CALCULATE ELEMENTS WEIGHTS BASED ON AGE

Age	Diagnostics Time	Penalty
1st	To 120 days	Not applicable
2nd	From 121 days to 150 days	10%
3rd	From 151 days to 180 days	20%
4th	From 181 days to 210 days	30%
5th	From 211 days to 240 days	40%
6th	From 241 days to 270 days	50%
7th	From 271 days to 300 days	60%
8th	From 301 days to 330 days	70%
9th	From 331 days to 360 days	80%
10th	From 360 days	90%

D. Strategy 4 - Questions' Popularity

In Strategy 4, the calculation of questions weights considers both the weight and the question's popularity, as detailed in Equation 2, which considers several factors, including: (*i*) question's popularity, obtained by the ratio between α_x (amount of occurrences of the question x) and n (amount of diagnostic sets selected), (*ii*) probability of identifying an RC is represented by β_{RCi}, which is obtained by the ration between the RC weight and the sum of RCs weights associated with the category analyzed, and (*iii*) amount of occurrences of question x in the diagnostic set of an RC, denoted as $\alpha_{RCi,x}$.

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

Like Strategy 2, Strategy 4 considers both completed and frustrated diagnostics. However, as can be seen in Equation 2, the questions' weights is obtained by the sum operation between the question's popularity and the sum of the probabilities of RCs in which the question belongs to the diagnostic sets. Finally, the computed value for this operation is divided by 2. As a result, a value between 0 and 1 that represents the calculated weight to the question is obtained. It is worth mentioning that each question can be present only once in each set of diagnostics. Thus, if $\alpha_{RCi,x}$ is equal to 0, the probability of identifying the evaluated RC is not used in the sum. On the other hand, if $\alpha_{RCi,x}$ is equal to 1, the probability of identifying the evaluated RC is used in the sum and, therefore, it is considered to calculate the questions weights.

Using the detailed information from Tables I and II, it is possible to calculate the questions weights using Equation 2. It is important to notice that RCs and categories weights are calculated using Strategy 2 and are ordered as detailed previously. However, the questions' weights are calculated according to the equation above and ranked as follows: (*i*) P_5 has weight 0.30204, (*ii*) P_4 and P_7 have weight 0.26327, (*iii*) P_1, P_2, and P_8 have weight 0.23265, (*iv*) P_3 and P_9 have weight 0.20204, (*v*) P_6 has weight 0.16122, and, finally, (*vi*) P_{10} has weight 0.14122. As in Strategy 2, the module selects the category C_1. However, in Strategy 4, the selected question is P_5.

V. CASE STUDY

Our improved solution has been applied to an emulated corporative IT infrastructure. In our case study, all four strategies proposed have been employed. Therefore, it is possible to analyze the results generated by each strategy and observe trends among them. It is important to emphasize beforehand that during the case study the failed activity, the IT infrastructure, and the RC identified have not been changed. That allows the fair comparison and evaluation of the resulting diagnostic workflows. Following, we demonstrate the correct operation of our solution, highlighting the main features, such as flexibility, interactivity, and knowledge reuse.

The scenario we analyze is based on a company that provides many sorts of Web-based services, *e.g.*, web hosting, email, e-commerce. The current IT infrastructure to support these services consists of two servers named *DB Server* and *Web Server*. While *DB Server* serves only for the purpose of the database service, the *Web Server* hosts all the remaining ones. However, aiming to deal with the growing number of customers, as well as to improve the availability and performance of offered services, the company choose to deploy two new servers. As a first step, the CIs associated with the Web hosting service shall be migrated to a new *Hosting Server*. After that, the CIs involved in the spam filter and email exchange will be migrated to the new *Mail Server*.

In order to perform these changes, an RFC is designed to deploy new servers and subsequently proceed with the migration of the selected services. As explained before, the defined RFC undergone a refinement process, conducted by Change Management System. This process results in the creation of a CP. During the deployment of the this CP, the system detects a failure in the activity *Verify Hosted Sites*. The state of the IT infrastructure when this failure was reported is detailed in Figure 2. Rectangles represent CIs as specified by CIM, and dependencies between them are depicted by arrows. Continuous arrows represent direct dependencies, whereas dashed ones map dependencies on services that are hosted on remote computer systems and, therefore, require a network connection to provide communication.

All CIs of the IT infrastructure are represented in Figure 2. Thus, when a PR is issued, the system calculates, through the dependencies, the subset of CIs that may have failed. In this case study, the failed activity (*Verify Hosted Sites*) is performed in CI *Hosted Sites*. Therefore, the selected dependent CIs

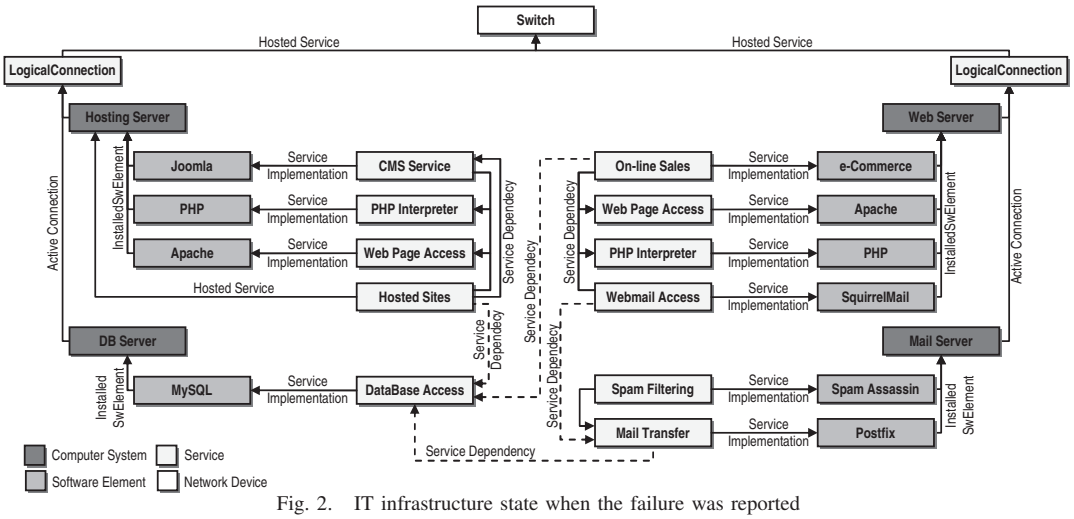

Fig. 2. IT infrastructure state when the failure was reported

are: (*i*) the services, *Database Access*, *Web Page Access*, *PHP Interpreter*, *CMS Service* and *Logical Connection*, (*ii*) the applications, Joomla, PHP, Apache, and MySQL, (*iii*) the computer systems, *DataBase* and *Hosting Server*, and, finally, (*iv*) the network device, *Switch*.

Afterwards, the Input Processor module identifies the categories associated with each selected CI. The identified categories and the weights calculated by each strategy, as of the first interaction, are shown in Table III. It is important to notice that *Service* is a Level 1 category and has two Level 2 categories associated with it, *Web Page Server* and *DataBase*. Moreover, notice that our proposed strategies add the weight of inferior level categories to the associated larger level ones. Therefore, the weight of *Web Page Server* and *DataBase* is added to the weight of *Service*.

TABLE III
IDENTIFIED CATEGORIES AND WEIGHTS CALCULATED BY STRATEGIES

Category	Level	Weights			
		Str. 1	Str. 2	Str. 3	Str. 4
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

In order to select a question for the first interaction, the system identifies which Level 1 category has the largest weight. As shown in Table III, Strategy 1 selects the category *Software*, as opposed to other strategies that first select the category *Network*. Although they identify the same category, Strategies 2, 3, and 4 may still select different questions. This is due to the fact that each strategy uses different criteria to calculate the weights of questions. Moreover, when identifying a Level 1 category, the related questions may be associated with any dependent category. Thus, when using Strategy 1, the questions included in the selection should be linked to the categories *Software*, *Web Server*, or *DB Server*. Similarly, when using Strategies 2, 3, or 4, the selected questions should be related to the categories *Network*, *Services*, or *Devices*.

A closer look in relation to the weights calculated by the proposed strategies is also important. Since Strategy 1 does not consider frustrated diagnosis, the weights calculated are larger as compared to other strategies. This allows the selection of questions associated with RCs that present large amounts of completed diagnosis, although many of those have been frustrated. Moreover, this approach delays the identification of RCs recently added, because of the discrepancy between the weights of a new RC and the ones identified repeatedly over time. In order to deal with this discrepancy, the Strategy 2 penalizes RCs for every frustrated diagnosis.

Strategy 2 does alleviate the discrepancy between the weights, but it still does not take into account the age of diagnosis. Therefore, RCs that have been identified in a large number of failures in the distant past, have their questions selected in many interactions until their weights are reduced. This generates a large number of additional interactions and consequent waste of time. For this reason, Strategy 3 considers the age of logs prioritizing recent diagnoses. In Table III, when comparing the calculated weights for the category *Software*, it is possible to observe a significant reduction between the values obtained by Strategies 2 and 3. This reduction is a result of the penalty applied based on the age of logs.

The diagnostic workflows generated by our solution, after executing the steps detailed in Section IV, are shown in Figure 3. The workflows generated by Strategies 1, 2, 3, and 4 are respectively depicted in Figure 3 (a), (b), (c), and (d). The rectangles with gray background represent the questions answered by the *Question Verifier*, while the rectangles with white background represent the questions answered by the operator. Below each question, there is a box containing information about the selection of the question.

As shown in Figure 3, Strategy 4 identified the RC with fewer interactions, as opposed to Strategy 1, which took the largest number of interactions. This improvement is due to

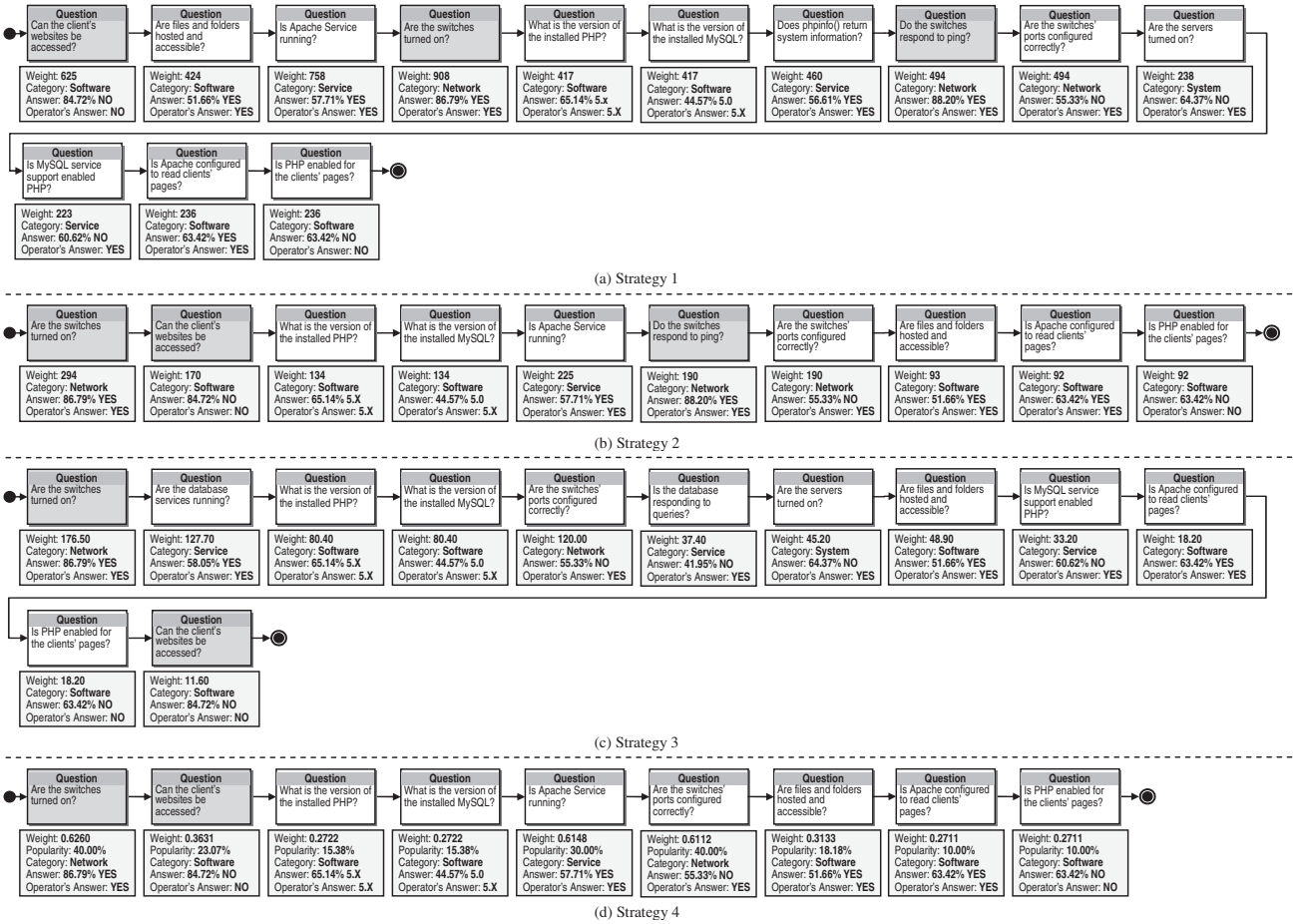

Fig. 3. Diagnostic workflows created using different strategies

using different criteria to calculate the weights. While Strategy 1 considers only completed diagnoses, Strategy 4 considers the entire history of diagnoses, as well as the popularity of each question. This allows the early selection of questions associated with many RCs, eliminating RCs that have questions in common and that cannot be identified with fewer interactions.

In a way, the proposed strategies already prioritize questions related to various RCs by adding to the weight of the question the weight of all associated RCs. However, there are situations where two questions have the same or very approximated weights. When this happens, Strategy 4 is the only one that prioritizes the questions contained in several sets of diagnosis, thus increasing the weight of the most popular questions. This can be seen in the diagnostic workflow of Strategy 4, which selected the question "*Are the switches' ports configured correctly?*" rather than "*Do the switches respond to ping?*", as did Strategy 2. It is important to emphasize that Strategies 2 and 4 employ the same approach for the selection of categories, but different criteria for selecting questions.

Another aspect refers to the amount of Level 1 categories selected. Strategies 1 and 3 changed 9 times the category used for selecting questions. On the other hand, Strategies 2 and 4 switched between categories only 5 times. This amount of changes reflects in the total number of interactions between the system and the operator, because for each change in the category at least one more question is selected. Besides that, the question "*Does phpinfo() return system information?*" is selected only by the Strategy 1. Similarly, questions "*Are the database services running?*" and "*Is the database responding to queries?*" are selected only by Strategy 3. This happens for two main reasons: (*i*) the large number of diagnoses penalized in Strategy 3 results in a flat range of the weights, thus questions not necessary in other strategies happen to be selected, and (*ii*) the large amount of frustrated diagnoses that are not penalized in Strategy 1 results in larger weights for RCs that have a heterogeneous history and thus prioritizes questions that do not accurately reflect past experiences. Therefore, the employment of these criteria as performed by Strategies 1 and 3 lead towards the selection of specific questions and to a larger amount of category level switching.

Another observation is needed on the question "*Can the client's websites be accessed?*". In Strategies 1, 2, and 4, this question is among the earliest selected. However, when employing Strategy 3, this question is selected only at the end of the diagnosis process. This is due to the fact that there are many diagnoses that get penalized for their age, as well as a large number of frustrated diagnoses. Therefore, the weight of this question is severely reduced, resulting in its late selection.

Remarkably, despite considering the same infrastructure and the same failure, all four proposed strategies came out with different diagnostic workflows. Such workflows differ not only in the number of questions, but also in the selected questions themselves. Nevertheless, the diagnosis process identifies always the same RC, namely: "*The PHP configuration does not allow the use of language in user's sites.*".

VI. Final Considerations and Future Work

Although the Change Management and Problem Management processes have received great attention in recent years, solutions developed so far do not identify the root cause of failures occurred in IT changes. To address this issue, in a previous work [8], we proposed an interactive solution for identifying the root cause of failures. In such solution, the Diagnosis System selects a question to be answered by the operator's system. Nevertheless, the factor used to select questions did not allow representing the full history of diagnoses. Then, in this paper, we propose improvements in the previous solution, as well as four novel strategies that use different criteria, such as popular or recent questions.

The results obtained demonstrate the correct identification of RCs using the improved solution, as well as the flexibility of diagnostic generated, the reuse of operator's knowledge, and system compatibility with the standards used by organizations. Moreover, the four proposed strategies generated different diagnostic workflows. These workflows differ in the questions used, and in quantity of questions. Furthermore, the improved solution allowed diagnosing root cause of failures that cannot be diagnosed automatically, such as human errors and misconfiguration of devices. Finally, the modular structure of our proposed solution allows organizations to adapt the improved solution to the special needs of their IT environments.

From the results of our research, we have the following recommendations for IT operators. Strategy 1 is recommended to history with a small amount of records. In these cases, it is important not to apply penalizations, otherwise the range among the questions weights would be reduced, requiring more interactions. Strategy 2 is recommended for bulky and recent histories. This strategy penalizes questions for each diagnosis frustrated. Such penalty prioritizes questions according to full history of diagnoses, as well as allows questions associated with new RCs to be used quickly. For logs that include at least 10 months, Strategy 3 is recommended. Thus, the age factor is used only when it truly improves the results, avoiding additional computing. Finally, Strategy 4 is recommended for data sets with a great amount of popular questions. Such questions are prioritized and, hence, fewer interactions are needed. In general, Strategies 2 and 4 showed better results considering scenarios with random historical data sets.

As future work, we plan to: (*i*) explore new criteria for the selection of questions, such as, false positives rates, false negative rates, and confidence, (*ii*) investigate the use of CIM classes (*e.g.*, dependencies, checks, and actions) in order to identify root causes, even when there are no logs of diagnoses, (*iii*) automate root cause identification of certain failures types, such as software failures and lack on minimum hardware requirements, and (*iv*) extend the process to identify root causes for other scopes, for example, applying the same methods to manage incidents in IT projects and service support.

References

[1] J. Sauvé, 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.

[2] R. Rebouças, J. Sauvé, A. Moura, C. Bartolini, and D. Trastour, "A Decision Support Tool to Optimize Scheduling of IT Changes," in *Proceedings. IFIP/IEEE International Symposium on Integrated Network Management (IM)*, Munich, Germany, May 2007, pp. 343–352.

[3] OGC, "Information Technology Infrastructure Library (ITIL)," 2011, available at: <http://www.itil-officialsite.com/>. Accessed: January 2011.

[4] OGC, *Information Technology Infrastructure Library: Service Operation Version 3.0*. London, UK: Office of Government Commerce, 2007.

[5] Y. Diao, H. Jamjoom, and D. Loewenstern, "Rule-Based Problem Classification in IT Service Management," in *Cloud Computing, 2009. IEEE International Conference on*, sept. 2009, pp. 221 –228.

[6] G. S. Machado, W. L. C. Cordeiro, F. F. Daitx, C. B. Both, L. P. Gaspary, L. Z. Granville, C. Bartolini, A. Sahai, D. Trastour, and K. Saikoski, "Enabling Rollback Support in IT Change Management Systems," in *Proceedings. IEEE/IFIP Network Operations and Management Symposium (NOMS)*, Salvador, Brazil, April 2008, pp. 347–354.

[7] M. Agarwal and V. Madduri, "Correlating failures with asynchronous changes for root cause analysis in enterprise environments," in *Dependable Systems and Networks (DSN), 2010 IEEE/IFIP International Conference on*, july 2010, pp. 517–526.

[8] R. dos Santos, J. Wickboldt, R. Lunardi, B. Dalmazo, L. Granville, L. Gaspary, C. Bartolini, and M. Hickey, "A solution for identifying the root cause of problems in it change management," in *Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on*, may 2011, pp. 586 –593.

[9] A. Keller, J. Hellerstein, J. Wolf, K. Wu, and V. Krishnan, "The CHAMPS system: Change management with planning and scheduling," in *Proceedings. IEEE/IFIP Network Operations and Management Symposium (NOMS)*, vol. 1, Seoul, Korea, 2004, pp. 395–408.

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

[11] S. Hagen and A. Kemper, "Towards solid IT Change Management: Automated detection of conflicting IT change plans," in *Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on*, may 2011, pp. 265 –272.

[12] M. Rahmouni and C. Bartolini, "Learning from past experiences to enhance decision support in IT change management," in *Network Operations and Management Symposium (NOMS), 2010 IEEE*, april 2010, pp. 408 –415.

[13] 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.

[14] C. Bartolini and C. Stefanelli, "Business-driven IT management," in *Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on*, may 2011, pp. 964 –969.

[15] C. Bartolini, M. Salle, and D. Trastour, "IT service management driven by business objectives An application to incident management," in *Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP*, april 2006, pp. 45 –55.

[16] DMTF, "CIM - Common Information Model," 2008, available at: <http://www.dmtf.org/standards/cim>. Accessed: Novembro 2009.

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

[18] W. Cordeiro, G. Machado, F. Andreis, A. D. Santos, C. Both, L. Gaspary, L. Granville, C. Bartolini, and D. Trastour, "ChangeLedge: Change Design and Planning in Networked Systems based on Reuse of Knowledge and Automation," *Computer Networks*, pp. 2782–2799, 2009.