Anatomy of a Fix: Analyzing Solution Patterns in Public IT Incident Reports

Insights from Postmortems on Mitigations and Fixes in Production Systems

Supervisors: Diomidis Spinellis[1], Eileen Kapel[1]

[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 25, 2025

Name of the student: Martin Georgiev
Final project course: CSE3000 Research Project
Thesis committee: Prof.dr.ir. D. (Diomidis) Spinellis PhD, Eileen Kapel, Benedikt Ahrens

Abstract

This study examined common remediation strategies by analysing publicly available IT incident reports. A six-category taxonomy ("Software Fix", "Rollback", "Traffic Switch", "Hardware/Infrastructure Repair or Operation", "Self-Resolved", and "Undisclosed/Not Specified") was developed to classify implemented solutions. Subsequently, a corpus of 1268 recent public incident reports sourced from the VOID community database was collected, from which the solution description of each report is classified utilising a prompt-based approach with the LLaMA3.3-70B-Versatile large language model (LLM). The LLM classifier demonstrated substantial agreement (with Cohen's $\kappa = 71.4\%$, Macro F1 = 80.6%) with manual annotations on a ground truth subset of 127 reports. The primary findings revealed that a significant majority (76%) of the reports do not disclose specific technical solutions. Among reports with identifiable fixes, software fixes (5.5% of total) were the most common. Exploratory analysis also showed a statistically significant but small relationship between solution category and incident duration. This research highlights the utility of LLMs for analysing incident reports and powering AIOps and underscores the need for improvement in incident reporting.

1 Introduction

Software-driven services play a crucial role in critical sectors such as finance, healthcare, and retail [1]. These services are inevitably prone to incidents for various reasons. Consequently, such incidents and a lack of effective incident management can affect service quality, customer trust [2], and business reputation [3].

To prevent incidents and improve service reliability, organisations often conduct post-incident analyses, commonly documented in incident reports or postmortems. These documents capture valuable information on the causes, impacts, and resolutions of operational failures [4]. However, these incident reports are typically written in free-text format. Therefore, the quality of the data represented in these reports is varied and inconsistent. Inconsistencies in the quality and structure of incident report data present a significant natural language processing (NLP) challenge, especially as modern advancements in artificial intelligence (AI) increasingly rely on data to power AIOps (artificial intelligence for IT operations), with numerous studies advocating for improvement in training data quality [5; 6; 7]. That is because, although not always stated directly, the effectiveness of AIOps platforms implicitly depends on the availability of high-quality, well-structured data [5]. Consequently, overcoming the challenge of analyzing these varied reports to systematically identify resolution patterns is crucial. Doing so would provide insights into prevalent resolution techniques which can subsequently inform better system design, guide the development of automated mitigation/resolution strategies, and improve operational training.

Furthermore, when studying incident reports, previous work has examined the complete life cycle of incidents [8] but despite that, few methods are tailored to recognise, extract and analyse "solution statements" (e.g. "rolled back to version X.X", "patched the configuration file") across heterogeneous report formats and organisational styles.

This need for further investigation into implemented solutions aimed at resolving incidents, along with the need for improvement of AIOps incident data quality, directly motivate the present study. In this paper, 1268 real-world incidents in large online systems are systematically analysed through their publicly available postmortems. This research aims to systematically classify and analyze them in the context of operational issues in information technology to review the primary solutions employed to address the issues. More concretely, it is structured around the following research questions:

- **RQ1:** How can solution descriptions be effectively identified and extracted from incident reports with nonstandardised structures?
- **RQ2:** What classification scheme or taxonomy best categorises the types of solutions found in incident reports?
- **RQ3:** What is the frequency distribution of different solution categories in the incident reports analyzed?

The methodology employed web scraping techniques to gather reports, followed by natural language processing and text classification methods, utilising an LLM and prompt engineering, to categorise the described solutions. The study culminated in a statistical analysis of the classified solutions to identify the most common remediation patterns. The main contributions are a practical analysis of how common faults are typically resolved based on a range of real-world incident reports as well as a framework for classification of incident solutions.

2 Background and Related Literature

This section reviews foundational work in IT incident management, relevant classification taxonomies, and the application of AI/ML and NLP techniques to incident report processing across various domains, with a focus on identifying gaps in solution classification automation.

2.1 Manual IT Incident management

In the area of IT incident management, significant work has been done examining the causality chain of incidents, along with the incident's characteristics and correlations with recently done changes to the system. It is important to mention that in the context of systems and software engineering, the term "changes" is defined as the modification of an existing application comprising additions, changes, and deletions [9]. Additionally, an incident is defined as an unplanned interruption to a service or a reduction in the quality of a service at a specific time [9]. Y. Wu et al. manually examined 161 postmortem reports from Ant Group and identified prevalent incident characteristics and mitigation strategies [10]. Some

works also directly provided software solutions aimed at simplifying and automating the process of managing information security incidents [11]. Furthermore, Zhao et al. have conducted a broader study on the entire life cycle of incidents and provide important findings such as effectiveness of incident detection options [8]. More importantly, they presented a detailed analysis of mitigation strategies, their commonality, their time efficiency, and the impact of mitigation strategy failures. Zhao et al. also argued that different strategies are most common for different primary causes and that some solutions (namely those that employ multiple resolution strategies and hotfixes) are more time-costly, primarily because of higher amounts of manual efforts [8]. These aforementioned works, as well as most of the literature, focus on exclusively change-induced incidents. Consequently, the discussion of incident management typically extends only as far as incident detection prior to and following the change, with limited attention given to the exploration or analysis of incident mitigation strategies.

2.2 Common Taxonomies

Several taxonomies have been proposed in the literature, as well as in industrial settings, to categorise aspects of IT incidents, with varying degrees of focus on incident characteristics versus their implemented solutions.

Guo and Wang briefly discussed the prevalence of the ITIL (Information Technology Infrastructure Library) [12] framework as an industry standard in the world of IT service management [13]. ITIL itself does not define a detailed "solution taxonomy", but when it is implemented in practice, common resolution categories are sometimes recorded as "resolution codes" [14]. ITIL's taxonomy is therefore operational and process-driven. Due to its nature, it is also very coarse-grained and the resolution categories (e.g. patch, training, hardware fix) vary between organisations and tools, and ITIL itself offers little analytical guidance on choosing among them. While ITIL focuses primarily on incident resolution from a service management perspective, other taxonomies have attempted to address the classification of the incidents themselves in more technical contexts.

For instance, Truong and Halper highlighted the need for better understanding of IoT (Internet of Things) data analytics to more adequately deal with potential incidents in these systems [15]. Consequently, they also suggested a classification taxonomy that focuses mainly on the context of each incident. In it, incidents are classified based on where in the software stack the incident occurred, what capacity of the overall system it affected, and when it occurred. However, this taxonomy remains relatively vague and lacks robust, well-defined categories that strike an effective balance between generalisation and granularity, limiting its practical applicability in heterogeneous systems and their postmortem reports. In addition, it is concerned with categorising the incidents themselves rather than their mitigation solutions used.

In contrast, Zhao et al. [8] and Y. Wu et al. [10] proposed taxonomies that are notably more precise and practically applicable for the purposes of the present study. Both offer similar classification schemes for incident mitigation strategies that provide a more effective mix of generalisation and specificity, making them better suited. These two taxonomies form the basis of the classification framework used in this paper, with only minor adjustments to fit the specific context of this work.

2.3 Use of ML and NLP in Incident Report Processing

Substantial research has been done, including systematic reviews, on the topic of using ML/NLP for text classification of reports in domains such as aviation [16; 17] and healthcare [18; 19]. In particular, R. Dillon et al. explored the difficulties and consequences of applying AI/ML and NLP in the context of learning from near-miss events' reports [20]. They did so by compiling information on past work utilising these techniques in analyzing incident datasets and then providing recommendations for usages as well as highlighting that the main challenge is training these models to reliably distinguish meaningful near misses from inconsequential events.

2.4 IT Incident Management via ML and NLP

To date, only a small number of studies have directly explored the area of employing modern AI approaches to classify incidents and their solutions based on postmortem reports. For example Sufi [21] proposed a novel approach to extracting cyber threat features from textual descriptions of cyber events by harnessing the capabilities of GPT. They concluded that such a paradigm can be beneficial for critical decision making during a cyber attack, make the identification of incidents more precise, and improve the understanding of attack patterns. In a broader context, Chen et al. conducted quantitative analysis of cloud service incidents at Microsoft, identified key insights into the difficulty of incident management and suggested an AIOps framework aimed at increasing the efficiency at different incident managements phases [22].

3 Methodology

In order to achieve the aforementioned objective of the research, a quantitative approach was employed. It combines automated data extraction, text classification, and statistical analysis to investigate the types of solutions commonly used to fix primary faults. The main stages of the work were as follows.

3.1 Data Acquisition and Formatting

The primary data sources for this study was the VOID community database [23]. The dataset used in this study consists of the latest 1500 report entries available as of May 23, 2025. This amount was deemed a sufficient collection of accessible reports from the targeted source. It is aimed to capture a broad range of publicly documented IT incidents to allow for a comprehensive exploration of common solution patterns. The volume was determined to be manageable within the project's time frame and resource constraints for automated analysis and subsequent manual validation. For the primary analysis, a subset of 1301 reports published within the last four years (2022–2025) was selected. This temporal filtering ensures the findings reflect modern and relevant operational practices and technologies, which have notably

evolved in recent years, partly influenced by accelerated digital transformation and shifts in working patterns spurred by events such as the COVID-19 pandemic. Of these, 33 reports were discarded due to being dead/invalid entries in the database. Out of the remaining 1268, a subset of 127 reports (10%) were chosen via a pseudo-random number generator with fixed seed and were utilised to create a manually labeled ground truth set. The data was gathered via a custom Python-based "Incident Report Management Script" (IRMS). This script manages several key stages of the data pipeline through an interactive command-line interface, leveraging a modular architecture composed of specialised processing and scraping components. It collects the data by calling the backend API endpoint of the VOID database, eliminating the need to scrape the front-end. After collection, the reports get formatted into JSON objects containing all the necessary and relevant information about them.

3.2 Solution Classification

The core of the paper is concerned with classifying the extracted solution texts into relevant categories in order to aggregate types of employed solutions and analyse them later on.

Taxonomy Development

One of the key objectives of this research was to establish a classification scheme that effectively categorises the types of solutions found in the dataset of reports. The development of this scheme was an iterative data-driven process, informed by a combination of existing literature, standards in the IT incident management field, and qualitative analysis of a sample of the gathered reports.

First, an initial review of academic works related to IT incident management and postmortem analysis was done to identify existing taxonomies or classification frameworks for incident causes and resolutions (as mentioned in subsection 2.2). Based on this review and common IT operational practices, a preliminary set of broad solution categories was formulated. These initial categories aimed to cover common high-level actions such as code-related fixes, configuration changes, infrastructure operations, and data interventions. Next, a randomly selected subset of 20 incident reports from the dataset were manually examined. The goal was to understand how applicable the initial draft of classes was to the data at hand. Based on this analysis, the initial taxonomy was iteratively refined. This involved merging categories that were too similar, removing categories not relevant to the dataset, adding categories for frequently observed solution types, and rephrasing category names to improve clarity and robustness.

This process resulted in the six-category taxonomy (SW: "Software Fix", RB: "Rollback", TS: "Traffic Switch", HW: "Hardware/Infrastructure Repair or Operation", SR: "Self-Resolved", ND: "Undisclosed/Not Specified") that is central to this study. For each category, a precise definition was established, along with examples of typical keywords and phrases encountered in the reports that would indicate a solution belonging to that category. The detailed description of the taxonomy, including the comprehensive definitions and keywords is the subject of **RQ1** and is presented in subsection 4.1.

Classifier model

To perform this classification, a publicly available pre-trained model was used to classify the texts based on a given taxonomy without specific tuning of hyperparameters for the dataset at hand. The chosen model was Meta's LLaMA3.3-70B-Versatile which is optimised for a wide range of natural language processing tasks. The model was accessed through a commercially available third-party API provided by GROQ [24]. The IRMS interacted with this API by sending the incident report text and receiving the predicted solution category. To ensure as deterministic outputs as possible and enhance the reproducibility of the classification process, the temperature parameter of the large language model (LLM) was set to 0.

Although recent studies such as the one done by Bucher and Martini [25] demonstrate that fine-tuned small LLMs can outperform large generative models in highly specific classification tasks such as sentiment analysis from news articles, the applicability of these findings must be examined with respect to the task at hand. The present work involves extracting and classifying relatively objective technical descriptions of incident mitigation actions from postmortem reports. These texts are typically factual, procedural, and minimally subjective in nature. Unlike sentiment classification or nuanced opinion mining (where subjective interpretation is central), the current task deals with identifying well-defined categories (e.g., rollback, patch deployment) based on relatively formulaic language. This reduces the need for domain-specific fine-tuning, as the linguistic cues involved are typically straightforward. Finally, utilising a fine-tuned model involves additional complexity such as data curation, compute resources for tuning, and accounting for risk of overfitting. More specifically, curating a balanced training dataset is infeasible for this paper, since the preliminary set is already highly unbalanced, and manually extracting a balanced subset would not only leave too few samples, but would also be impractical given the 10-week timeline.

The prompt used as the system prompt for the classifier was manually developed in accordance to common prompt engineering practices[26]. Each of the six solution categories was clearly defined, accompanied by examples and lists of "Keywords/Phrases often associated". This aims to provide the LLM with concrete anchors. Emphasis was placed on identifying the primary fix and distinguishing it from preventative measures or secondary actions. The prompt mandated a specific two-part output: a quoted solution snippet which aims to serve as justification for the chosen class followed by a single categorical letter. The entire prompt can be seen in Appendix A

To assess the effectiveness and reliability of the LLM in classifying incident report solutions, a comprehensive performance evaluation was conducted. This evaluation used a manually annotated ground truth dataset comprising 127 incident reports. The LLM's predicted categories for these reports were compared against the ground truth labels. The following metrics were calculated to quantify the classifier's

performance:

- ***Absolute Accuracy and Cohen's Kappa:*** Done to capture overall performance and agreement with the ground truth set. These metrics were interpreted in conjunction with class-specific measures due to potential class imbalances.
- ***Per-Class True Positive (TP), False Positive (FP), and False Negative (FN) Rates:*** Derived from the confusion matrix, these rates were examined for each individual solution category to understand the classifier's strengths and weaknesses at a more granular level.
- ***Per-Class Precision, Recall, and F1-Score:*** Precision measured the proportion of correctly classified reports per category, recall measured the proportion of actual instances correctly identified, and F1-score provided the harmonic mean of both metrics for balanced performance assessment.

3.3 Statistical Analysis

Following the classification of incident reports into the developed solution taxonomy (as described in section 3.2, a series of statistical analyses were performed to quantify the findings and explore patterns within the data. These analyses primarily utilised Python libraries such as Pandas [27; 28], Seaborn [29], NumPy [30], and SciPy [31; 32]

- ***Frequency Distribution of Solution Categories:*** The primary analysis involved calculating the frequency distribution of the classified solution types. For each of the solution categories, both the absolute number of incidents falling into that category and its relative frequency (percentage of the total analyzed reports) were computed. These distributions were then visualised using bar charts to clearly illustrate the prevalence of different solution strategies, directly addressing the main research question regarding commonly employed fixes.
- ***Word Frequency Analysis within Quoted Solutions (per Category):*** To gain deeper insight into each solution category, a word frequency analysis was conducted on the "quoted solution" snippets extracted by the LLM during the classification process. For each solution category, common NLP preprocessing steps (e.g., lowercasing, stop word removal, punctuation removal) were applied to the texts. Subsequently, the frequency of individual words (unigrams) was calculated. This aimed to identify distinctive terminology and action verbs associated with each solution type, further validating and distinguishing the categories.
- ***Relationship between Solution Category and Incident Duration:*** To examine the potential statistical relationship between an incident's duration and the implemented solution, a compilation of every report that has disclosed its incidents duration was created. From there, key characteristics per class such as mean and median duration per class, along with standard deviations were calculated. Additionally, the eta coefficient (η), a measure of association appropriate for relationships between nominal categorical variables and continuous variables was calculated to quantify the proportion of variance in the duration of an incident explained by the class membership of its solution.

$$\eta = \sqrt{\frac{\sum_{k=1}^{K} n_k(\bar{x}_k - \bar{x})^2}{\sum_{i=1}^{N}(x_i - \bar{x})^2}}$$

Unlike correlation coefficients such as Pearson's r, which assume linear relationships, the eta coefficient makes no assumptions about the functional form of the relationship and can capture non-linear associations between variables.

4 Results

This section details the empirical findings of the study. It first describes the solution classification taxonomy developed from the analysis of incident reports. After that is the evaluation of the automated classification methodology, followed by the frequency distribution of the identified solution categories within the studied corpus.

4.1 A Taxonomy for Classifying Incident Report Solutions

The taxonomy used in this paper was adapted from Zhao et al. [8] and Y. Wu et al. [10], as mentioned in subsection 2.2 and section 3.2. It was employed due to its practical applicability, balance between granularity and generalisation, and direct focus on categorising mitigation actions based on real-world postmortem analyses of change-induced incidents. The full taxonomy is listed in Table 1

This taxonomy serves as the foundation for classifying solution texts extracted from incident reports and allows for a structured quantitative analysis of the frequency and distribution of each mitigation strategy. It should be noted that these classes are derivative of Zhao et al. and Y.Wu et al.'s previous work with some key differences:

- The classification used in this paper omits the "Fallback-/Degraded service" category that is present in the work of Zhao et al. but also missing in the taxonomy used by Y.Wu et al. This is because any fallback-type mitigation can reasonably be ascribed to either "Infrastructure Change" or "Software Fix", as fallback mechanisms typically rely on pre-configured infrastructure behavior or code-level logic designed to degrade cleanly under some failure conditions.
- The "Hybrid" class from Zhao et al.'s work was also omitted from this paper's taxonomy. This was to avoid loss of granularity and unwanted ambiguity in thresholding. The definition of what constitutes a "Hybrid" case can very which can consequently complicate classification logic and risk inconsistency in labeling. Not including a "Hybrid" class prioritises action specificity over response sequence bundling.
- Neither of the aforementioned previous studies provide a specific category for vague or undisclosed fixes. Based on an initial manual review of the dataset, the introduction of an "Undisclosed/Not Specified" category was

Table 1: Incident resolution categories and associated keywords

Resolution Category	Definition	Keywords/Phrases Often Associated
SW: Hotfix/ Software Fix	The solution involved deploying a new piece of code, a patch, or a hotfix to correct a software bug or issue. This implies a change to the existing codebase that moves it forward, rather than reverting.	"deployed a patch", "released hotfix", "fixed the bug in code", "merged fix and deployed", "applied code change"
RB: Rollback	The solution involved reverting a recent deployment (code, configuration, or feature flag) to a previously known good state. This refers specifically to undoing a recent change.	"rolled back deployment", "reverted to previous version", "undid change", "restored prior configuration", "disabled feature flag that was just enabled"
TS: Traffic Switch	The solution involved rerouting user traffic or data flow, typically by failing over to a standby system/region, bypassing a problematic component or environment, or updating load balancer/DNS settings.	"failed over to secondary", "switched traffic to new cluster", "rerouted users", "updated DNS to point to", "diverted traffic"
HW: Hardware/ Infrastructure Repair or Operation	The solution involved rebooting, scaling, and/or isolating faulty containers or physical machines. It could also include performing operational actions on infrastructure services (e.g., restarting a core service, scaling resources, rebooting hosts).	"replaced faulty disk/server", "rebooted host machine", "restarted critical infrastructure service", "scaled up VM resources", "network device repair"
SR: Self-Resolved	The issue resolved itself without direct manual intervention by the reporting team, often due to transient conditions. Issues in this category often stem from scheduled maintenances.	"issue self-resolved", "transient network glitch cleared", "system automatically recovered", "scheduled maintenance"
ND: Undisclosed/Not Specified	This category includes cases where: (1) the specific fixing action is not clearly stated, omitted, or too vague; (2) the resolution involved an external dependency fixed by a third party; or (3) no identifiable fix is mentioned.	"vendor resolved upstream issue", "no specific action detailed", "monitoring and investigation ongoing"

necessary to account for a significant portion of incident reports that lacked sufficient detail about the mitigation strategies employed. As a result, categorising these entries under specific technical solution types would require an undue degree of inference and risk introducing noise or misclassification into the dataset.

4.2 Automated Classification Accuracy

The model LLaMA3.3-70B-Versatile was tasked with classifying solution descriptions extracted from incident reports into one of six classes (SW, RB, TS, HW, SR, ND). Its performance was then evaluated against a manually annotatted set of 127 incident reports

The overall accuracy achieved was 87.4%. To account for agreement occurring by chance, Cohen's Kappa coefficient (κ) was calculated, yielding a value of 71.4%, indicating substantial agreement between the LLM's predictions and the ground truth set. Furthermore, the classifier achieved a macro F1 score of 80.6%.

A detailed breakdown of the classification performance is presented in the confusion matrix (Figure 1) and per-class metrics (Table 2).

4.3 Solution Strategies Analysis

The primary analysis focused on the prevalence of each of the six defined solution categories. Figure 2 illustrates the frequency of each solution type, and Table 3 provides the absolute and relative counts. As most reports had little to no explanation regarding the implemented solution, the classes are largely imbalanced. Consequently, class ND has been omitted in Figure 2 for visualisation and clarity purposes.

Table 2: Classification Metrics

	SW	RB	TS	HW	SR	ND
TP	7	2	5	5	5	87
FP	2	0	1	0	0	13
FN	1	1	1	0	12	1
Prec.	0.78	1.00	0.83	1.00	1.00	0.87
Recall	0.88	0.67	0.83	1.00	0.29	0.99
F1	0.82	0.80	0.83	1.00	0.46	0.93

In terms of word frequency, the top 5 most frequently occurring content words, excluding common function words (e.g., 'a', 'the', 'of'), also known as stop words, are presented below in Table 4

Lastly, the existence of a statistical relationship between an incident's solution and its duration was examined. In order to do that, an additional filtering for report entries that do not contain the incident's duration needed to be done. Additionally, the class of undisclosed solutions (ND) was excluded from this analysis and comparative testing due to its undefined nature. Out of the 1268 total reports, only 1156 had disclosed the duration of their respective incidents. Following

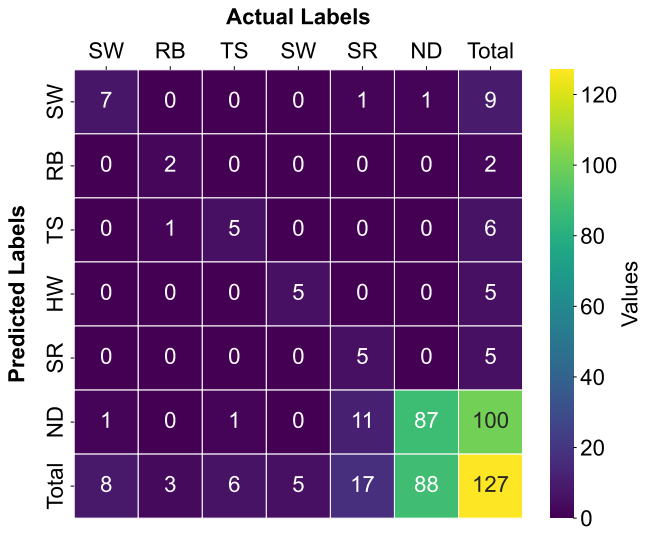

Figure 1: Confusion matrix of predicted (rows) against actual (columns) labels

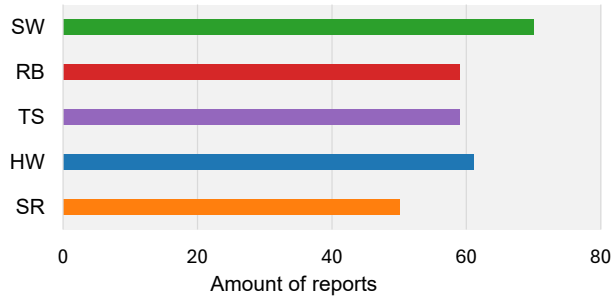

Figure 2: Bar graph of report distribution among the five solution classes, ND class excluded

Table 3: Distribution of reports by class

Category	Count	Percentage
Reports in class SW	70	5.52%
Reports in class RB	59	4.65%
Reports in class TS	59	4.65%
Reports in class HW	61	4.81%
Reports in class SR	50	3.94%
Reports in class ND	969	76.42%
Total	**1268**	**100.00%**

that, a total of 876 belonged to the ND class and were therefore excluded. The analysis was ultimately done over 280 entries spanning the SW, RB, TS, HW, and SR classes. The breakdown of key information regarding the durations associated with each class can is presented in Table 5. The data show classes SW and HW have significantly longer mean durations (1300.1 and 1657.5 minutes) compared to RB, TS, and SR, although their medians (210 and 283 minutes) are only

Table 4: Top 5 Most Frequent Non-stop Words by Class

Category	Top 5 Words (with Frequencies)
SW	fix (40), deployed (24), issue (21), monitoring (9), identified (8)
RB	back (31), change (27), issue (20), rolled (19), configuration (10)
TS	traffic (44), temporarily (30), rerouted (29), different (8), region (7)
HW	issue (18), engineers (10), manually (9), mitigated (9), traffic (9)
SR	maintenance (35), scheduled (34), completed (33), resolved (7), issue (6)
ND	monitoring (366), fix (364), implemented (354), results (330), issue (88)

modestly higher. In contrast, classes RB and TS have similar moderate durations (medians of 185 and 199, and means of 316 and 357 respectively) with relatively smaller variability (standard deviations of 396 and 457), while the SR class has a relatively low median (60) but moderate mean (613.8).

A formal test confirmed these differences are statistically significant. A Kruskal–Wallis nonparametric ANOVA on ranked durations yielded $p \approx 1.05 \cdot 10^{-6}$, allowing to reject the null hypothesis of equal distributions across solution types. However, the effect size was small: ≈ 0.215 ($\eta^2 \approx 0.046$), meaning only about 4.6% of the total variance is explained by solution category.

Table 5: Descriptive statistics regarding duration in minutes for each class

Class	Count	Mean	Med	Std Dev	Min	Max
SW	63	1300.10	210	3582.78	4	25920
RB	57	356.98	185	456.70	8	2100
TS	57	316.23	199	395.52	7	2441
HW	57	1657.54	283	3522.07	22	19616
SR	46	613.76	60	2074.05	1	10080

5 Responsible Research

This research was carried out according to principles of responsible research practice. To allow for replicability, a replication package containing the used code, the dataset, the ground truth set, and all environmental specifications will be made available [33]. All data utilised consists of publicly available incident reports sourced from established online repositories such as the VOID Community database. Data collection scripts were designed to respect website terms of service and employed rate limiting to minimise server load. The study focused solely on the technical content of these reports pertaining to incident causes and solutions, and no attempt was made to collect or analyze personally identifiable information.

The methodology, including data preprocessing, classification techniques, and analytical procedures, is described

transparently to facilitate reproducibility and critical review. The random sampling for the ground truth set was done via a seeded randomness generator to ensure reproducibility. In addition, the visualisations presented in section 4 utilise colorblind-friendly color palettes in order to ensure accessibility for readers with color vision deficiency.

It is important to acknowledge that no AI classifier is perfect and misclassifications could influence the precise frequency distribution of solution types. LLMs are often "black boxes" in the sense that understanding why a particular category was chosen for a given data instance can be difficult. This lack of transparency can be a concern if significant decisions are made based on the LLM's output without understanding the underlying reasoning. The requirement for the LLM to output a quoted solution can be viewed as a localised explainability, providing evidence for the classification which can then be used by a human to verify the classification decision. Additionally, LLMs require significant computational resources, which has an environmental impact [34]. This study employed LLaMA3.3-70B-Versatile, utilising pre-existing scaled infrastructure. By avoiding the need to train a model from scratch, the study significantly reduced its potential environmental footprint.

In a more broader context, an emerging concern in the literature is an over-reliance on automated tools [35], and more specifically AI models, without sufficient human oversight, which can potentially lead to new issues or a general "de-skilling" of human operators [36]. This research aims to provide insights that *inform* human understanding and can *assist* in the development of better tools and practices. It is **not** intended to suggest a complete replacement of human expertise in incident analysis or resolution but rather to augment it with data-driven patterns.

Finally, the usage of AI/LLMs throughout the conducting of the study is described in Appendix B

6 Discussion

This study aimed to develop a practical taxonomy for classifying solutions in IT incident reports, evaluate an automated approach for classification of said reports' solutions, and determine the prevalence of different solution types. The results presented provided valuable insights into these areas, while also highlighting the complexities associated with examining unstructured public incident data.

6.1 The Taxonomy and its Utility

The overall analysis required balancing granularity of categories against practical constraints. The six-category taxonomy in this study was developed based on Zhao et al. [8] and Wu et al. [10], but some nuanced classes were intentionally omitted, as mentioned in subsection 4.1. This streamlining improves consistency but means certain strategies are lumped into broader groups. The findings echo this. In practice, the "Self-Resolved" (SR) class includes both routine scheduled maintenance and incidental recoveries, which are qualitatively different but grouped together. The decision to introduce the "Undisclosed/Not Specified" (ND) class was also validated by the data as it was the most represented class by far (76% of all reports). It allowed for accommodating for the significant number of reports lacking detailed resolution information.

The word frequency analysis (Table 4) further supported the distinctiveness of the disclosed solution categories (SW, RB, TS, HW, SR), with terms like "deployed" for software fixes (SW), "rolled" and "back" for rollbacks (RB), "traffic" and "rerouted" for traffic switches (TS), "engineers" and "manually" for hardware repairs (HW), and "maintenance" and "scheduled" for self-resolved (SR) incidents largely aligning with their definitions. Interestingly, the word frequencies of reports in the ND class were significantly higher than those in the other classes. The observed higher level of homogeneity is likely because, in the case of deliberate lack of disclosure about the way the issue was resolved, the subsequent reports likely follow a pre-defined format that superficially describes the characteristics of the issue without elaborating on the solution. This also aligned with the lack of specificity in the most common words for the class.

6.2 Performance of Automated Classification

The LLaMA3.3-70B-Versatile model demonstrated substantial agreement (Cohen's Kappa = 71.4%) with manual annotations and a strong macro F1 score (80.6%), indicating its effectiveness in automating the classification of solution types. The performance is weaker than the one reported by Sufi [21] regarding using LLMs for identifying characteristics of attacks from historical cyber incident reports. However, that study was specifically aimed at optimising model accuracy for the task, whereas the present work prioritised generalisability and minimal fine-tuning. As such, the stronger performance observed in Sufi's study is to be expected. Nevertheless, the overall accuracy of 87.4% is promising for applying such models to larger datasets. The confusion matrix (Figure 1) revealed generally good performance, especially for well defined actions like hardware or infrastructure changes (HW). The lower recall in "Self-Resolved" incidents suggests the LLM struggled with these cases and misclassified them regularly as "Undisclosed/Not Specified". This is likely due to the nature of self-resolved incidents and the subtle linguistic cues that differentiate them from an omitted resolution statement.

6.3 Prevalence of Solution Strategies

The most frequent solution category overall was ND, accounting for 76% of reports, reflecting that many of the reports omit explicit fixes or describe only monitoring. The prevalence of undisclosed solutions highlights a significant characteristic of public incident reporting: the frequent lack of detailed disclosure regarding the specific technical actions taken to resolve incidents. This observation aligns with the decision to include class ND in the proposed taxonomy and suggests that some companies, for reasons such as confidentiality or reputational risk, may limit the technical depth of their public-facing postmortems. However, this finding should be interpreted with caution, as companies with less transparent disclosure practices may publish a disproportionate number of reports.

Among the five disclosed-solution classes (SW, RB, TS, HW, SR), software fixes were the most common (70 incidents, 5.5%) and self-resolved (SR) the least common (50 incidents, 3.9%). Rollbacks (RB), traffic switches (TS), and hardware repairs (HW) each appear in roughly 4.7–4.8% of cases. Therefore, among the actionable solutions, implementing a patch or code update is slightly more common. This finding may reflect the complexity of modern software systems, where issues often require targeted code changes rather than simpler operational maneuvers. The results contrast with the findings of Zhao et al. [8], who observed that rollbacks were the most common mitigation strategy. Importantly, that study examined exclusively change-induced incidents, potentially biasing the distribution toward fixes like rollbacks that directly reverse recent changes. Overall, the reported mitigation solutions are relatively balanced. This reveals that the challenge the industry is facing lies not in a lack of diverse solutions but in the frequent non-disclosure in public reports.

6.4 Incident Duration and Solution Types

The findings of the exploratory analysis largely align with prior research on mitigation strategies. In particular, Zhao et al. found that solutions involving hotfixes or multi-step manual interventions tend to be more time-consuming due to greater manual effort [8]. Consistent with that, classes SW and HW have the longest resolution times on average. Conversely, simpler, pre-planned strategies like rollbacks or traffic failsafes (RB and TS) have shorter times, which is to be expected for more automated fixes. The discrepancy between mean and median in SW and HW suggests a heavy "long tail", meaning a few complex cases took extremely long to resolve. This may reflect that some hotfixes required significant debugging or that certain hardware fixes encountered supply or scheduling delays.

6.5 Implications of the Study

The high prevalence of "Undisclosed/Not Specified" solutions suggests a critical challenge for researchers and practitioners relying on public incident reports for AIOps model training or understanding operational patterns. Additionally, it also creates barriers for cross-organisational learning and knowledge transfer. Without access to comprehensive incident resolution data, organisations must rely primarily on internal knowledge and trial-and-error approaches, perpetuating suboptimal response strategies across the industry. Lastly, even though the employed taxonomy was effective for the reports containing detailed solutions, the overwhelming number of undisclosed mitigation strategies limits the generalisability of the quantitative findings to the broader set of all publicly reported incidents. The potential link between solution type and incident duration deserves further investigation, as understanding these relationships could help in predicting resolution times or prioritising certain types of fixes based on urgency.

7 Limitations and Future Work

While this study provides valuable insights into common solution types documented in public incident reports, several limitations should be acknowledged, which also point towards directions for future research.

7.1 Limitations Regarding Data Collection

The primary dataset was drawn exclusively from the VOID community database. Although it is a rich resource, relying on a single aggregator may not capture the full diversity of publicly available incident reports hosted on individual company blogs not added to the database or other repositories/-databases.

It should also be mentioned that the nature of publicly available incident reports is subject to inherent reporting bias. It is reasonable to assume that larger companies, due to their scale and resources, are more likely to have comprehensive infrastructure for incident handling and are therefore more inclined to publish detailed postmortems. Due to the fact that the findings are based only on publicly available reports, generalising to all reports should be done with caution as the types of solutions common in highly secure, proprietary, or non-web-facing systems might differ.

7.2 Reporting Characteristics

The analysis focused on reports from the last three years to capture contemporary practices. While this ensures relevance, it cannot capture more long-term trends in solution types or reporting methodologies that might have evolved over a more extended period. Additionally, given how broad and varied IT incidents can be, the snapshot of 1500 (1268 used for primary analysis) reports, although appropriate for the scope of this study, may still be too limited to support firm pattern emergence, or commonality among solution types.

7.3 Classification Limitations

This study utilised a single large language model for automated classification. This approach does not allow for a comparative analysis of different models, which could provide insights into the general applicability of LLMs for incident report analysis or reveal model-specific biases. Every model comes with a certain amount of bias inherited from training data which can potentially manifest in specific error patterns (e.g. mistaking two particular classes for one another) which might not be as prevalent or present at all in other models. Furthermore, due to the use of an LLM model, the quantitative findings of the research are extremely dependent on its performance. As such, generalisations drawn from the results should take into account the limitations of the method used.

7.4 Taxonomy Limitations

The developed taxonomy aimed to strike a balance between granularity and generalisability, preserving the nuances of the data while also ensuring that the classes were broad enough to allow scalable observation and analysis. Despite efforts for objectivity, some degree of subjectivity can remain in defining the boundaries between categories. More broadly, any taxonomy is an abstraction. Incident mitigation can be multifaceted, involving a primary action accompanied by several secondary or supporting actions. Forcing a complex resolution into a single category inevitably leads to a loss of some detail and nuance.

7.5 Future Work

Future work could focus on expanding the dataset's scope and depth through more sophisticated data acquisition strategies. This includes incorporating a wider range of public sources - such as company engineering blogs, status pages, and other incident repositories, still adhering to ethical data collection practices of course. Additionally, with appropriate permissions, future studies could explore private, anonymised incident report datasets to compare findings with public data and mitigate publication bias. Collecting data over a longer historical period would also enable analysis of temporal trends in incident response strategies and reporting practices. To further enhance classification accuracy, techniques for identifying and extracting solution statements from full reports could be improved using advanced NLP models tailored for information extraction or question answering. Exploring multi-label classification approaches and hierarchical taxonomies might offer more granular categorisation. Finally, conducting comparative evaluations of different large language models such as various open-source or commercial APIs, as well as fine-tuned variants could yield valuable insights into their effectiveness for solution classification tasks.

8 Conclusion

This study set out to extract and analyze common solution patterns in public IT incident reports. Using a prompt-based LLM classifier, the primary fix action in each report was automatically identified and categorised. It was revealed that a single category—undisclosed solutions (ND)—dominates most reports, but among explicit solutions, hotfixes/software fixes (class SW) were the most frequent. Statistical tests showed that solution category is significantly associated with incident duration ($p<0.001$), yet the effect is small (only 4.6% variance explained). Results also revealed that some fix types (e.g., hotfixes or hardware repairs) tend to require longer work, while rollbacks and traffic switches were typically quick. This study highlights the potential value of clearer reporting standards. If incident reports consistently recorded and disclosed the exact solution employed to deal with the incident, future analyses could provide more precise insights. With respect to the AI/NLP community, this work shows promising results in utilising LLMs for report analysis, while also noting the importance of human reviews of the output of said models. The importance of combining automated pattern mining with expert knowledge should not be understated. Although trends emerge from data, human expertise remains crucial to interpret and act on them. Ultimately, while modern tools like LLMs can greatly enhance the speed and consistency of incident analysis, human oversight must remain the final arbiter.

References

[1] R. Alt, J. M. Leimeister, T. Priemuth, S. Sachse, N. Urbach, and N. Wunderlich, "Software-defined business," *Business & Information Systems Engineering*, vol. 62, no. 6, pp. 609–621, 2020.

[2] M. Lee and J. Lee, "The impact of information security failure on customer behaviors: A study on a large-scale hacking incident on the internet," *Information Systems Frontiers*, vol. 14, pp. 375–393, Apr. 2012.

[3] D. V. Eduardovich and Y. A. Vladimirovich, "Reputation risks through information security incidents," in *2016 IEEE NW Russia Young Researchers in Electrical and Electronic Engineering Conference (EIConRusNW)*, pp. 194–198, 2016.

[4] incident.io, "Post-mortem documents." https://incident.io/guide/learn-and-improve/post-mortem-documents, 2024. Accessed: 2025-05-14.

[5] L. Rijal, R. Colomo-Palacios, and M. Sánchez-Gordón, "AIOps: A Multivocal Literature Review," in *Artificial Intelligence for Cloud and Edge Computing* (S. Misra, A. Kumar Tyagi, V. Piuri, and L. Garg, eds.), pp. 31–50, Cham: Springer International Publishing, 2022.

[6] Y. Dang, Q. Lin, and P. Huang, "AIOps: Real-World Challenges and Research Innovations," in *2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*, pp. 4–5, May 2019. Journal Abbreviation: 2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion).

[7] S. Polisetty, "Training ai models: Preparing and managing ai algorithms for aiops," *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, vol. 9, no. 5, 2023.

[8] Y. Zhao, L. Jiang, Y. Tao, S. Zhang, C. Wu, Y. Wu, T. Jia, Y. Li, and Z. Wu, "How to Manage Change-Induced Incidents? Lessons from the Study of Incident Life Cycle," in *2023 IEEE 34th International Symposium on Software Reliability Engineering (ISSRE)*, pp. 264–274, Oct. 2023. ISSN: 2332-6549.

[9] ISO/IEC/IEEE, "Iso/iec/ieee international standard - systems and software engineering–vocabulary." Standard 24765:2017(E), 2017. pp. 1–541.

[10] Y. Wu, B. Chai, Y. Li, B. Liu, J. Li, Y. Yang, and W. Jiang, "An Empirical Study on Change-induced Incidents of Online Service Systems," in *2023 IEEE/ACM 45th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, pp. 234–245, May 2023. Journal Abbreviation: 2023 IEEE/ACM 45th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP).

[11] P. Khorev and V. Karpeeva, "Software Tools for Analyzing Information Security Incidents Based on Monitoring of Information Resources," 2022.

[12] Office of Government Commerce, *ITIL Core Books*. UK: Office of Government Commerce, 2010.

[13] W. Guo and Y. Wang, "An incident management model for SaaS application in the IT organization," pp. 137–140, 2009.

[14] Combodo, "itop 3.0 - itil incident management data model," n.d. Accessed: 2025-05-26.

[15] H.-L. Truong and M. Halper, "Characterizing Incidents in Cloud-Based IoT Data Analytics," vol. 1, pp. 442–447, 2018.

[16] C. Yang and C. Huang, "Natural Language Processing (NLP) in Aviation Safety: Systematic Review of Research and Outlook into the Future," *Aerospace*, vol. 10, no. 7, 2023.

[17] T. Madeira, R. Melício, D. Valério, and L. Santos, "Machine Learning and Natural Language Processing for Prediction of Human Factors in Aviation Incident Reports," *Aerospace*, vol. 8, no. 2, 2021.

[18] I. J. B. Young, S. Luz, and N. Lone, "A systematic review of natural language processing for classification tasks in the field of incident reporting and adverse event analysis," *International Journal of Medical Informatics*, vol. 132, p. 103971, Dec. 2019.

[19] C. R. Hölzing, S. Rumpf, S. Huber, N. Papenfuß, P. Meybohm, and O. Happel, "The Potential of Using Generative AI/NLP to Identify and Analyse Critical Incidents in a Critical Incident Reporting System (CIRS): A Feasibility Case–Control Study," *Healthcare*, vol. 12, no. 19, 2024.

[20] R. Dillon, P. Madsen, B. Holland, and D. Cao, "How AI Can Help Learn Lessons from Incident Reporting Systems," in *2024 IEEE Aerospace Conference*, pp. 1–15, Mar. 2024. Journal Abbreviation: 2024 IEEE Aerospace Conference.

[21] F. Sufi, "An innovative GPT-based open-source intelligence using historical cyber incident reports," *Natural Language Processing Journal*, vol. 7, p. 100074, June 2024.

[22] Z. Chen, Y. Kang, L. Li, X. Zhang, H. Zhang, H. Xu, Y. Zhou, L. Yang, J. Sun, Z. Xu, Y. Dang, F. Gao, P. Zhao, B. Qiao, Q. L. , D. Zhang, and M. R. Lyu, "Towards intelligent incident management: why we need it and how we make it," in *2020 Foundations of Software Engineering*, pp. 1487–1497, ACM, Nov. 2020.

[23] The Void Community, "The void postmortem database." https://www.thevoid.community/database, 2024. Accessed: 2025-04-25.

[24] "Groq console (groqcloud developer console)." https://console.groq.com. Accessed: 2025-05-30; provides developer dashboard, API key management, playground for models such as Qwen-3-32B, Llama4, Whisper, Mistral, and more.

[25] M. J. J. Bucher and M. Martini, "Fine-Tuned 'Small' LLMs (Still) Significantly Outperform Zero-Shot Generative AI Models in Text Classification," 2024. _eprint: 2406.08660.

[26] L. Giray, "Prompt Engineering with ChatGPT: A Guide for Academic Writers," *Annals of Biomedical Engineering*, vol. 51, pp. 2629–2633, Dec. 2023.

[27] W. McKinney, "Data structures for statistical computing in python," in *Proceedings of the 9th Python in Science Conference*, pp. 51–56, SciPy, 2010.

[28] T. p. d. team, "pandas-dev/pandas: Pandas," Sept. 2024.

[29] M. L. Waskom, "seaborn: statistical data visualization," *Journal of Open Source Software*, vol. 6, no. 60, p. 3021, 2021.

[30] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, pp. 357–362, Sept. 2020.

[31] Wes McKinney, "Data Structures for Statistical Computing in Python," in *Proceedings of the 9th Python in Science Conference* (Stéfan van der Walt and Jarrod Millman, eds.), pp. 56 – 61, 2010.

[32] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[33] I. Aldea, M. Georgiev, J. Rutkowski, A. Mureşan, and D. Bunschoten, "What can we learn from incident reports? - web scraper," June 2025.

[34] Y. Ding and T. Shi, "Sustainable LLM Serving: Environmental Implications, Challenges, and Opportunities : Invited Paper," in *2024 IEEE 15th International Green and Sustainable Computing Conference (IGSC)*, pp. 37–38, Nov. 2024. Journal Abbreviation: 2024 IEEE 15th International Green and Sustainable Computing Conference (IGSC).

[35] A. Klingbeil, C. Grützner, and P. Schreck, "Trust and reliance on AI — An experimental study on the extent and costs of overreliance on AI," *Computers in Human Behavior*, vol. 160, p. 108352, Nov. 2024.

[36] N. Kosmyna, E. Hauptmann, Y. T. Yuan, J. Situ, X.-H. Liao, A. V. Beresnitzky, I. Braunstein, and P. Maes, "Your brain on chatgpt: Accumulation of cognitive debt when using an ai assistant for essay writing task," 2025.

A Appendix A - System Prompt for Automatic Classification

```
You are an expert IT incident analyst tasked
with analyzing full incident reports (
postmortems) to identify and categorize the
primary solution implemented to resolve the
core issue.

**Your Goal:**
1.  Read the entire incident report provided below.
2.  Identify the section(s) or statement(s) that
    describe the *primary corrective action* or *
    implemented fix* that resolved the main problem
    .
3.  Based *only* on this identified solution,
    classify it into one of the following six
    categories.

**Focus:**
*   Prioritize the action that directly restored
    service or fixed the underlying fault.
*   Distinguish between immediate corrective
    actions and longer-term preventative measures
    or lessons learned (we are interested in the *
    immediate fix*).
*   If multiple actions were taken, identify the *
    most critical or impactful one* that led to
    resolution.

**Categories for the *Implemented Solution*:**

*   **A: Software Fix:** The solution involved
    deploying a *new* piece of code, a patch, or a
    hotfix to correct a software bug or issue. This
     implies a change to the existing codebase that
     moves it forward, rather than reverting.
    *   *Keywords/Phrases often associated:* "
    deployed a patch", "released hotfix", "fixed
    the bug in code", "merged fix and deployed", "
    applied code change"

*   **B: Rollback:** The solution involved
    reverting a recent deployment (code,
    configuration, or feature flag) to a previous
    known-good state. This is specifically about
    undoing a recent change.
    *   *Keywords/Phrases often associated:* "
    rolled back deployment", "reverted to previous
    version", "undid change", "restored prior
    configuration", "disabled feature flag that was
     just enabled"

*   **C: Traffic Switch:** The solution involved
    rerouting user traffic or data flow, typically
    by failing over to a standby system/region to
    bypass a problematic component or environment,
    changing load balancer configurations or DNS
    settings.
    *   *Keywords/Phrases often associated:* "
    failed over to secondary", "switched traffic to
     new cluster", "rerouted users", "updated DNS
    to point to", "diverted traffic"

*   **D: Hardware/Infrastructure Repair or
    Operation:** The solution involved rebooting,
    scaling and/or isolating faulty containers or
    physical machines. It could also include
    performing operational actions on
    infrastructure services (e.g., restarting a
    core service, scaling infrastructure resources,
     rebooting hosts).
    *   *Keywords/Phrases often associated:* "
    replaced faulty disk/server", "rebooted host
    machine", "restarted critical infrastructure
    service", "scaled up VM resources", "network
    device repair"

*   **E: Self-Resolved:** The issue resolved itself
     without direct manual intervention by the
    reporting team, often due to transient
    conditions. Issues in this category mostly stem
     from scheduled maintenances.
    *   *Keywords/Phrases often associated:* "issue
     self-resolved", "transient network glitch
    cleared", "system automatically recovered", "
    scheduled maintenance"

*   **F: Undisclosed/Not Specified/Other:** This
    category is for situations where:
    1.  The specific fixing action is not clearly
    stated, entirely omitted, or is too vague in
    the report.
    2.  The solution was an external dependency
    being fixed by a third party (and not detailed)
    .
    3.  If no clear single fixing action can be
    identified from the report.
    *   *Keywords/Phrases often associated:* "
    vendor resolved upstream issue", "no specific
    action detailed", "monitoring and investigation
     ongoing (if no fix)"
**Output Instructions:**

1.  First, on a new line, briefly **quote the exact
     key sentence(s) or phrase(s)** from the report
     that best describe the implemented solution
    you are classifying. If the solution is
    described across multiple sentences, try to
    pick the most concise and representative part.
    Limit this to 1-3 sentences. If no clear
    solution statement is found, write "
    SOLUTION_NOT_FOUND".
2.  Second, on the next new line, provide **ONLY
    the single letter** (A, B, C, D, E, or F)
    corresponding to the category for the
    identified solution.

**Example Output Format:**
Quoted Solution: "We successfully rolled back the
    new feature deployment to the previous stable
    version (v2.1.5)."
Category: B

Quoted Solution: "A patch was developed and
    deployed to address the null pointer exception
    in the billing module."
Category: A
```

```
Quoted Solution: "Traffic was immediately failed
    over to our DR site in us-west-2."
Category: C

Quoted Solution: "The issue appears to have been
    transient and resolved itself after
    approximately 15 minutes."
Category: E

Quoted Solution: "SOLUTION_NOT_FOUND"
Category: F

**Incident Report to Analyze:
```

B Appendix B - usage of AI

AI tools were also used as an aid to the writing process by creating initial drafts on paragraphs, improving text conciseness and sentence rephrasing. Additionally, AI was also used for automation of creating tables in LaTeX, as well as generating scripts for creating chart visualisations and data type conversions via Python. Finally, AI was also queried for explanations on different topics needed throughout the study such as navigating Excel functionalities, LaTeXand Python error outputs, machine learning terminology and providing summaries on package documentations. All content generated by AI was manually reviewed and thoroughly examined for quality assurance purposes and ensure proper functionality/correctness. At no point throughout the project was AI intended to be used as a substitute for critical thinking or human expertise.